Savitribai Phule Pune University

F. Y. B. C. A. (Science)

Semester-II

BCA - 127
Operating Systems Laboratory
Work Book

Name: ____

College Name:____

Roll No.:_______________________Division: ___________________________

Academic Year:_________________________________

Editors:

Assignment 1	Mr. Kamil Khan	Abeda Inamdar Senior College, Pune
Assignment 2	Mr. Praveen Kulkarni	New Arts, Commerce and Science College, Ahmednagar
Assignment 3	Mrs. Ashwini Patil	Dr.D.Y.Patil College,Pimpri, Pune
Assignment 4	Mrs. Veena Gandhi	Abeda Inamdar Senior College, Pune
Assignment 5	Mrs. Aparna Gohad	Abasaheb Garware College, Pune
Assignment 6	Mrs. Rupali Nankar	Modern College, Ganeshkhind, Pune
Assignment 7	Mrs. Madhuri Khadtare	Abasaheb Garware College, Pune
Assignment 8	Mrs. Veena Gandhi	Abeda Inamdar Senior College, Pune
Assignment 9	Mrs. Veena Gandhi	Abeda Inamdar Senior College, Pune

Compiled By:
Mrs. Veena Gandhi
Abeda Inamdar Senior College, Pune

Reviewed By:
Prof. Arun Gangarde,
New Arts, Commerce and Science College, Ahmednagar

Introduction

1. **About the work book:**
   This workbook is intended to be used by F.Y.B.C.A. (Science) students for the BCA127- Operating Systems Assignments in Semester–II. This workbook is designed by considering all the practical concepts / topics mentioned in syllabus.

2. **The objectives of this workbook are:**
   1) Defining the scope of the course.
   2) To bring the uniformity in the practical conduction and implementation in all colleges affiliated to SPPU.
   3) To have continuous assessment of the course and students.
   4) Providing ready reference for the students during practical implementation.
   5) Provide more options to students so that they can have good practice before facing the examination.
   6) Catering to the demand of slow and fast learners and accordingly providing the practice assignments to them.

3. **How to use this workbook:**
   The workbook is divided into nine assignments. Each assignment has three SETs. It is mandatory for students to complete SET A and SET B in given slot.

Instructions to the students

Please read the following instructions carefully and follow them.

1) Students are expected to carry this book every time they come to the lab for computer science practical.
2) Students should prepare oneself beforehand for the Assignment by reading the relevant material.
3) Instructor will specify which problems to solve in the lab during the allotted slot and student should complete them and get verified by the instructor. However student should spend additional hours in Lab and at home to cover as many problems as possible given in this work book.
4) Students will be assessed for each exercise on a scale from 0 to 5.

Not done	0
Incomplete	1
Late Complete	2
Needs improvement	3
Complete	4
Well Done	5

Instruction to the Instructors

1) Explain the assignment and related concepts in around ten minutes using whiteboard if required or by demonstrating the software.
2) You should evaluate each assignment carried out by a student on a scale of 5 as specified above by ticking appropriate box.
3) The value should also be entered on assignment completion page of the respective Lab course.

Instructions to the Lab administrator

You have to ensure appropriate hardware and software is made available to each student.

The operating system and software requirements on server side and also client side areas given below:

1) Server and Client Side - (Operating System) Linux(Ubuntu/Red Hat/Fedora) – any distribution
2) BASH shell for shell scripts

Table of Contents

Assignment Completion Sheet

Lab Course BCA-127 Operating Systems Laboratory			
Sr. No.	**Assignment Name**	**Marks (out of 5)**	**Teachers Sign**
1	Linux Installation		
2	Study of Linux desktop environment		
3	Linux Commands-I		
4	Linux Commands-II		
5	Linux Commands-III		
6	Vi Editor Commands		
7	Shell Programming - I		
8	Shell Programming - II		
9	Shell Programming - III		
Total (Out of 45)			
Total (Out of 15)			

Certificate

This is to certify that Mr. /Ms.____ .has successfully completed BCA-127 Operating Systems Laboratory course in year ________and his/her seat no. is __________He / she has scored __________ Marks out of 15.

Instructor	H.O.D. / Coordinator
Internal Examiner	External Examiner

Assignment 1

Aim: To Study about Linux operating system and installation of OS and application

Pre-requisite: Knowledge of basic operation in Linux operating system and partitioning

The student should read following topics before starting exercises.

Introduction

Linux is a popular version of the UNIX Operating System. It is open source as its source code is freely available. It is free to use. Linux was designed considering UNIX compatibility. Its functionality list is quite similar to that of UNIX.

Components of Linux System

Linux Operating System has primarily three components:

Kernel:

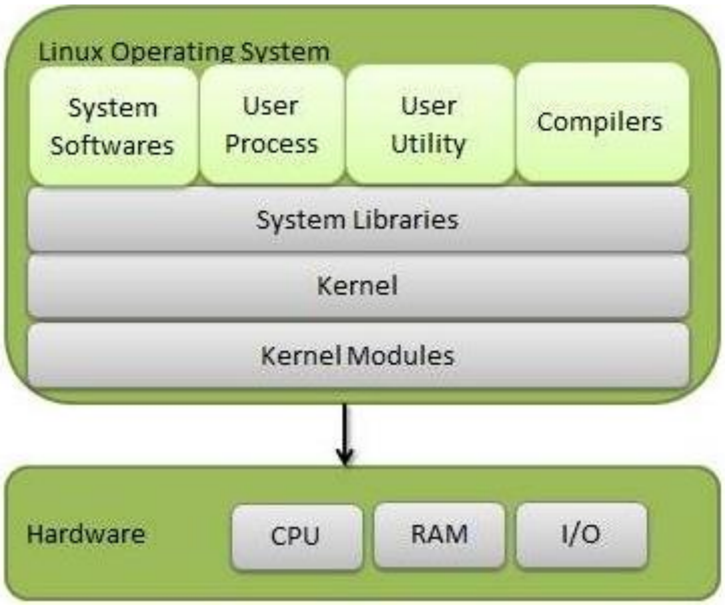
Kernel is the core part of Linux. It is responsible for all major activities of this operating system. It is consists of various modules and it interacts directly with the underlying hardware. Kernel provides the required abstraction to hide low level hardware details to system or application programs.

System Library:

System libraries are special functions or programs using which application programs or system utilities accesses Kernel's features. These libraries implement most of the functionalities of the operating system and do not require kernel module's code access rights.

System Utility:

System Utility programs are responsible for doing specialized, individual level tasks.



Components of Linux System

Important features of Linux Operating System

- Portable – Portability means software can works on different types of hardware in same way. A Linux kernel and application program supports their installation on any kind of hardware platform.
- Open Source – Linux source code is freely available and it is community based development project. Multiple team's works in collaboration to enhance the capability of Linux operating system and it is continuously evolving.
- Multi-User – Linux is a multiuser system means multiple users can access system resources like memory/ ram/ application programs at same time.
- Multiprogramming – Linux is a multiprogramming system means multiple applications can run at same time.
- Hierarchical File System – Linux provides a standard file structure in which system files/ user files are arranged.
- Shell – Linux provides a special interpreter program which can be used to execute commands of the operating system. It can be used to do various types of operations, call application programs etc.
- Security – Linux provides user security using authentication features like password protection/ controlled access to specific files/ encryption of data.
- Live CD/USB: Almost all Linux distributions have Live CD/USB feature by which user can run/try the OS even without installing it on the system.
- Graphical user interface (X Window System): People think that Linux is a command line OS, somewhere its true also but not necessarily, Linux have packages which can be installed to make the whole OS graphics based as Windows.
- Support's most national or customized keyboards: Linux is used worldwide and hence available in multiple languages, and supports most of their custom national keyboards.
- Application Support: Linux has its own software repository from where users can download and install thousands of applications just by issuing a command in Linux Terminal or Shell. Linux can also run Windows applications if needed.

Linux Distribution:

Linux distributions take the Linux kernel and combine it with other free software to create complete packages.

Here, are a few popular Linux Distributions (also called Linux Distro) –

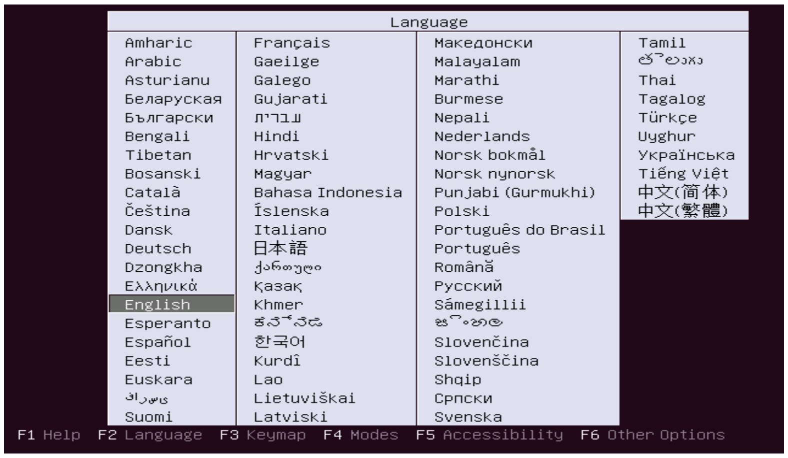
Linux Distribution	Name	Description
archlinux	Arch	This Linux Distro is popular amongst Developers. It is an independently developed system. It is designed for users who go for a do-it-yourself approach.
CentOS	CentOS	It is one of the most used Linux Distribution for enterprise and web servers. It is a free enterprise class Operating system and is based heavily on Red Hat enterprise Distro.

	Debian	Debian is a stable and popular non-commercial Linux distribution. It is widely used as a desktop Linux Distro and is user-oriented. It strictly acts within the Linux protocols.
	Fedora	Another Linux kernel based Distro, Fedora is supported by the Fedora project, an endeavor by Red Hat. It is popular among desktop users. Its versions are known for their short life cycle.
	Gentoo	It is a source based Distribution which means that you need to configure the code on your system before you can install it. It is not for Linux beginners, but it is sure fun for experienced users.
	LinuxMint	It is one of the most popular Desktop Distributions available out there. It launched in 2006 and is now considered to be the fourth most used Operating system in the computing world.
	OpenSUSE	It is an easy to use and a good alternative to MS Windows. It can be easily set up and can also run on small computers with obsolete configurations.
	RedHat enterprise	Another popular enterprise based Linux Distribution is Red Hat Enterprise.It has evolved from Red Hat Linux which was discontinued in 2004. It is a commercial Distro and very popular among its clientele.
	Slackware	Slackware is one of the oldest Linux kernel based OS's. It is another easy desktop Distribution. It aims at being a 'Unix like' OS with minimal changes to its kernel.
	Ubuntu	This is the third most popular desktop operating system after Microsoft Windows and Apple Mac OS. It is based on the Debian Linux Distribution, and it is known as its desktop environment.

Installing Ubuntu Operating System

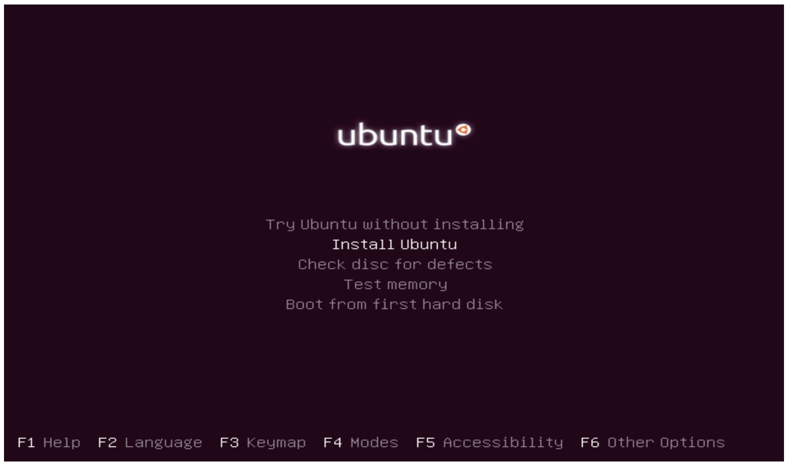
Download Ubuntu 16.04 from https://www.ubuntu.com/download, put the CD into the CD-ROM drive, change the boot sequence so that CD-ROM can boot first.
OR
If you want to **install Ubuntu from the USB flash Disk,** change the boot sequence according to the USB mass storage to boot first.

Press Enter to get a language screen and then select the language of your choice.



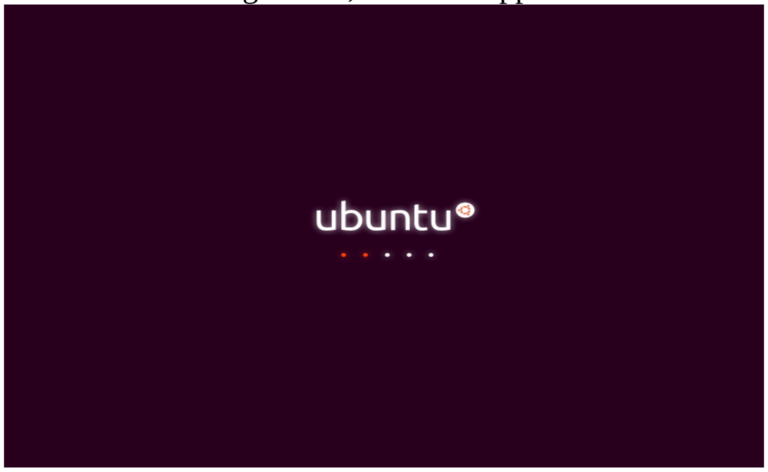
Install Ubuntu 16.04 – Installation language

For installing the Ubuntu 16.04, Select Install Ubuntu.


ubuntu
Try Ubuntu without installing
Install Ubuntu
Check disc for defects
Test memory
Boot from first hard disk
F1 Help F2 Language F3 Keymap F4 Modes F5 Accessibility F6 Other Options

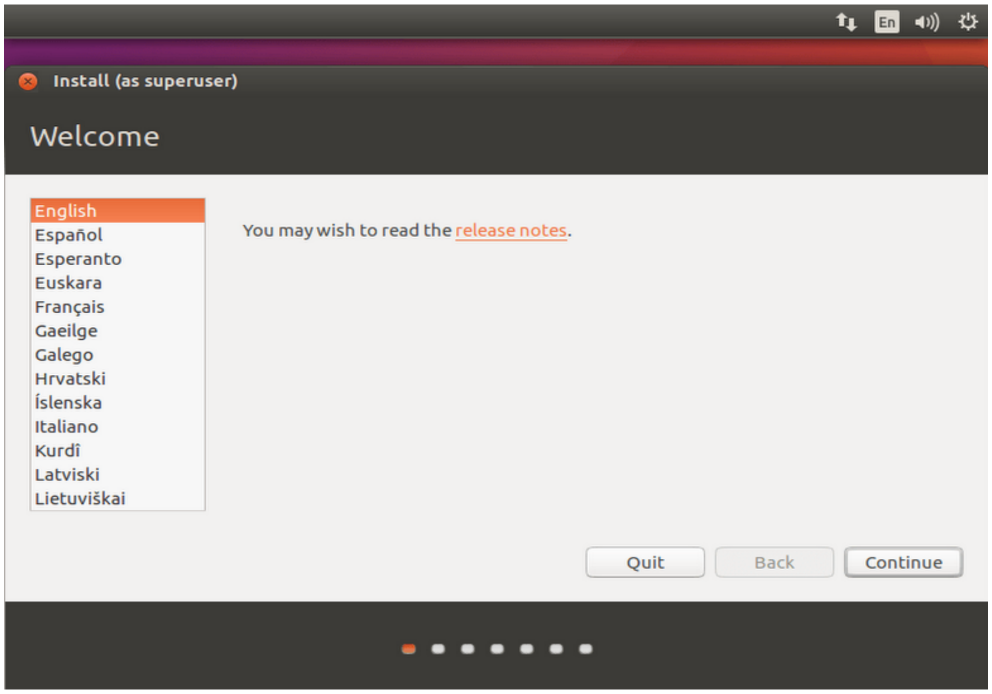
Install Ubuntu 16.04 – Menu

This is a starting screen, it will disappear in a minute



Install Ubuntu 16.04 – Starting Screen

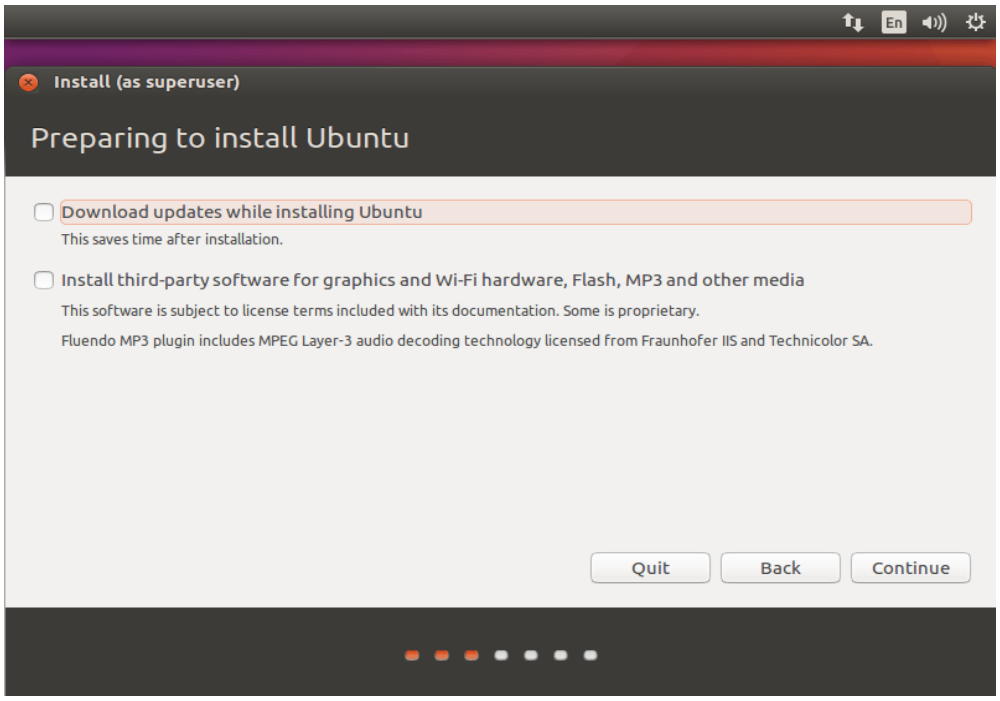
Click continue on the welcome screen.



You may wish to read the release notes.

Install Ubuntu 16.04 – Welcome Screen

You can either choose to install updates and other third-party software while installing Ubuntu 16.04 or leave as it is since it requires internet and installation may take an hour-long depends on the downloadable contents.



- [ ] Download updates while installing Ubuntu

  This saves time after installation.

- [ ] Install third-party software for graphics and Wi-Fi hardware, Flash, MP3 and other media

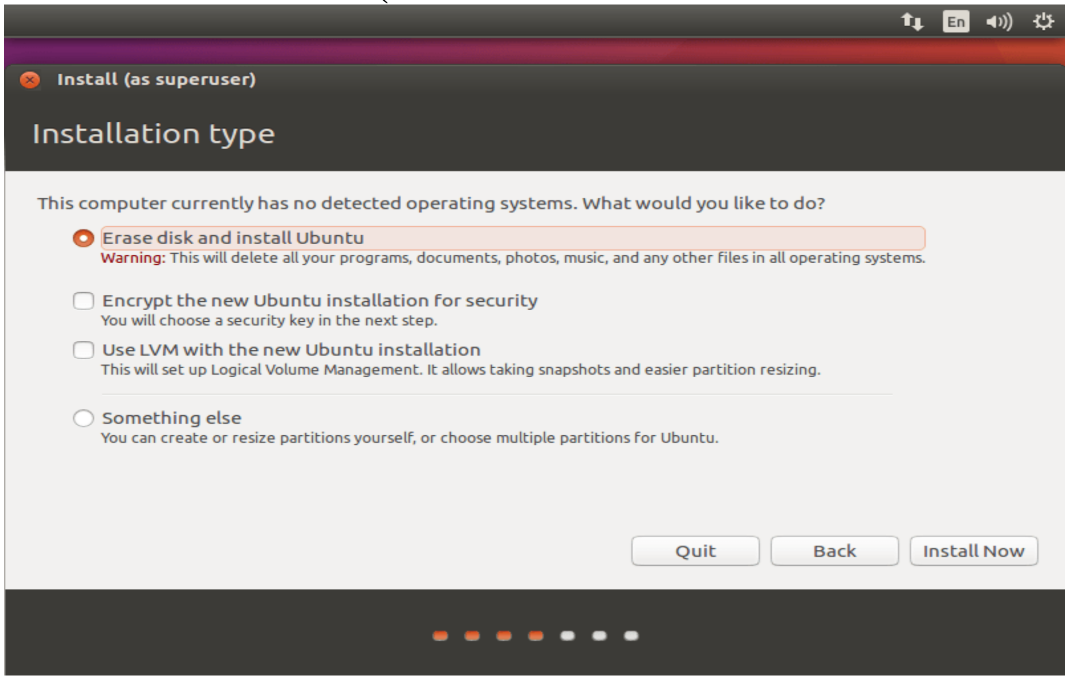
  This software is subject to license terms included with its documentation. Some is proprietary.

  Fluendo MP3 plugin includes MPEG Layer-3 audio decoding technology licensed from Fraunhofer IIS and Technicolor SA.

Ubuntu 16.04 – Preparing Installation

Next is the installation type, this installation on the fresh HDD so I have only two option in the installation type. Depend on the other OS on your HDD you will get more options. **Please chose any one of the methods.**

1. Erase disk and install Ubuntu (i.e. **it will format the entire drive and install the OS**).



Installation type

This computer currently has no detected operating systems. What would you like to do?

[✓ Erase disk and install Ubuntu]
Warning: This will delete all your programs, documents, photos, music, and any other files in all operating systems.

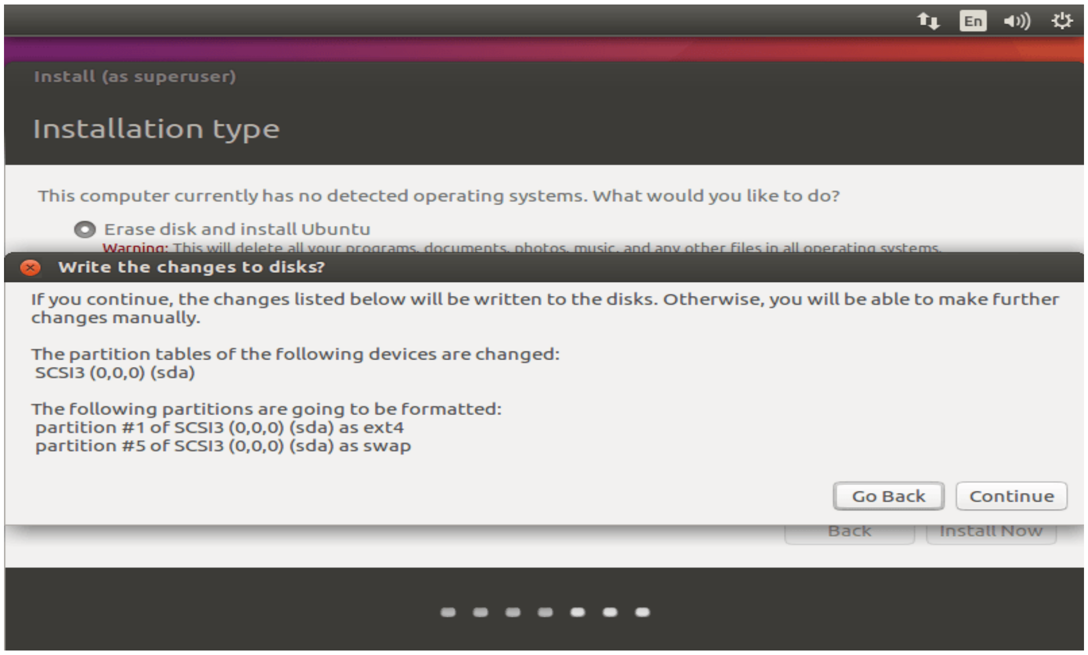
Encrypt the new Ubuntu installation for security
You will choose a security key in the next step.

Use LVM with the new Ubuntu installation
This will set up Logical Volume Management. It allows taking snapshots and easier partition resizing.

Something else
You can create or resize partitions yourself, or choose multiple partitions for Ubuntu.

Install Ubuntu 16.04 – Installation Type

Once you clicked on Install Now, the installer will ask you to confirm the auto partitioning. Click on continue.



Installation type

This computer currently has no detected operating systems. What would you like to do?

Erase disk and install Ubuntu
Warning: This will delete all your programs, documents, photos, music, and any other files in all operating systems.

Write the changes to disks?

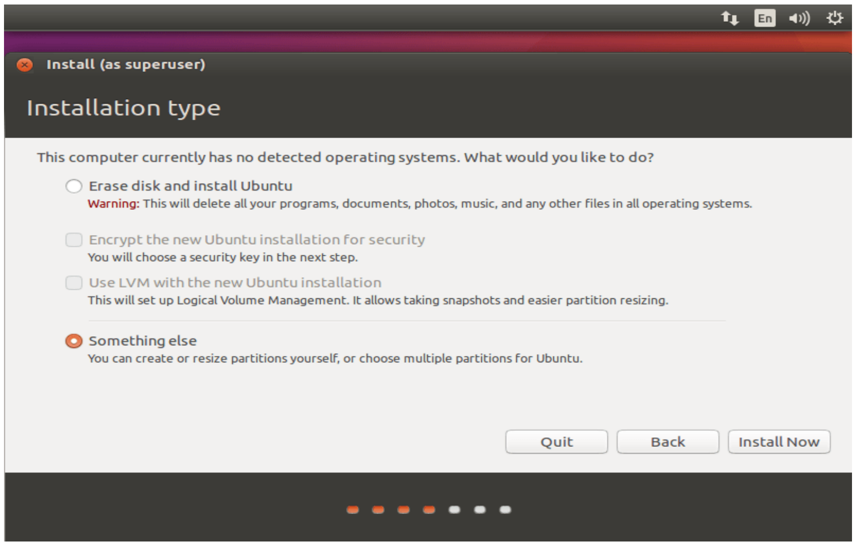
If you continue, the changes listed below will be written to the disks. Otherwise, you will be able to make further changes manually.

The partition tables of the following devices are changed:
SCSI3 (0,0,0) (sda)

The following partitions are going to be formatted:
partition #1 of SCSI3 (0,0,0) (sda) as ext4
partition #5 of SCSI3 (0,0,0) (sda) as swap

Install Ubuntu 16.04 – Erase Full Disk

2. Something else (i.e. **you can manually create the partition and install Ubuntu on your selected partition**), use this advanced mode if you are comfortable in partitioning your drives manually. Click on continue.


Install (as superuser)

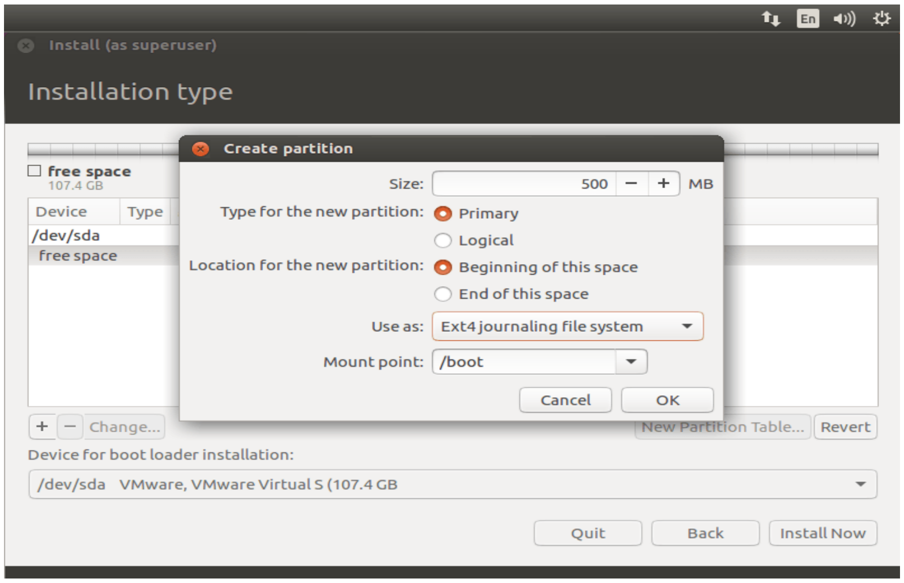
Installation type

This computer currently has no detected operating systems. What would you like to do?

- ( ) Erase disk and install Ubuntu
  Warning: This will delete all your programs, documents, photos, music, and any other files in all operating systems.
- [ ] Encrypt the new Ubuntu installation for security
  You will choose a security key in the next step.
- [ ] Use LVM with the new Ubuntu installation
  This will set up Logical Volume Management. It allows taking snapshots and easier partition resizing.
- [✓ Something else]
  You can create or resize partitions yourself, or choose multiple partitions for Ubuntu.

Install Ubuntu 16.04 – Installation Type

Once you clicked, you would get the following page where installer lists available hard disk. In my case I have one hard disk size of 100GB, to create a partition; click on New Partition Table to create an empty partition.
Since this is a new hard disk. Pop up will ask you to confirm, click on continue.
Partition scheme will be like below:
/boot – 500MB
/swap – 2048MB
/ – Remaining (99GB)
Select free space and click on the **+ sign** at the bottom to create partitions. Following shows for /boot partition.


Install (as superuser)

Installation type

free space
107.4 GB

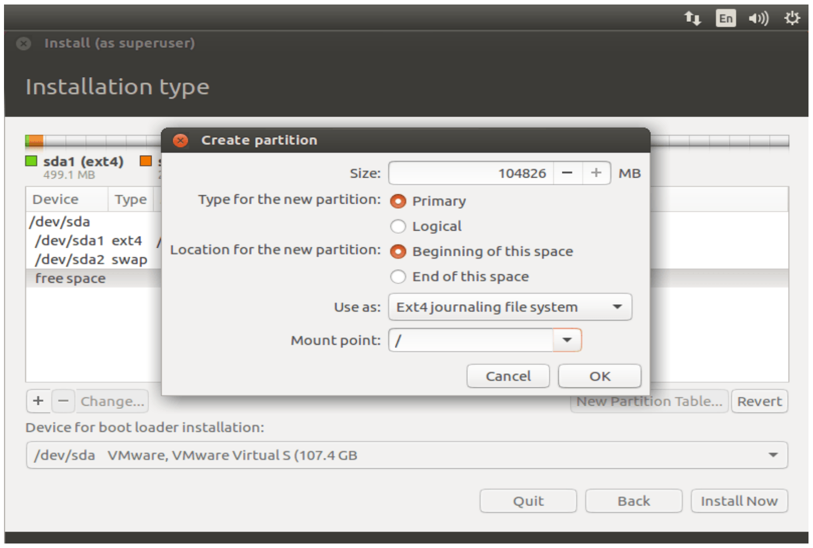
Device	Type
/dev/sda	
free space	

Create partition

Size: 500 MB
Type for the new partition: [✓ Primary] Logical
Location for the new partition: [✓ Beginning of this space] End of this space
Use as: Ext4 journaling file system
Mount point: /boot

Device for boot loader installation:
/dev/sda VMware, VMware Virtual S (107.4 GB

Install Ubuntu 16.04 – boot partition


Install (as superuser)

Installation type

sda1 (ext4) 499.1 MB

Device	Type
/dev/sda	
/dev/sda1	ext4
/dev/sda2	swap
free space	

Create partition

Size: 104826 MB

Type for the new partition: [✓ Primary] Logical

Location for the new partition: [✓ Beginning of this space] End of this space

Use as: Ext4 journaling file system

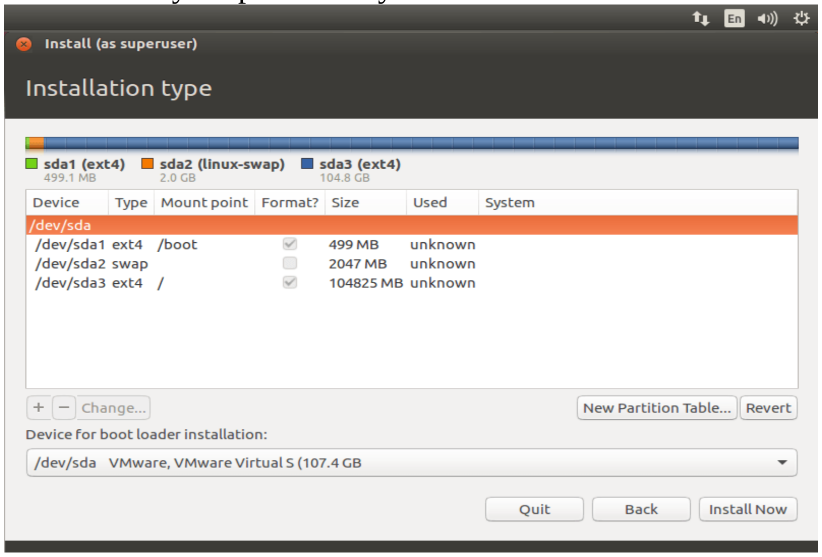
Mount point: /

Device for boot loader installation:

/dev/sda VMware, VMware Virtual S (107.4 GB

Following is for / (root) partition.
Install Ubuntu 16.04 – root partition

Review your partition layout and click on install now.


Install (as superuser)

Installation type

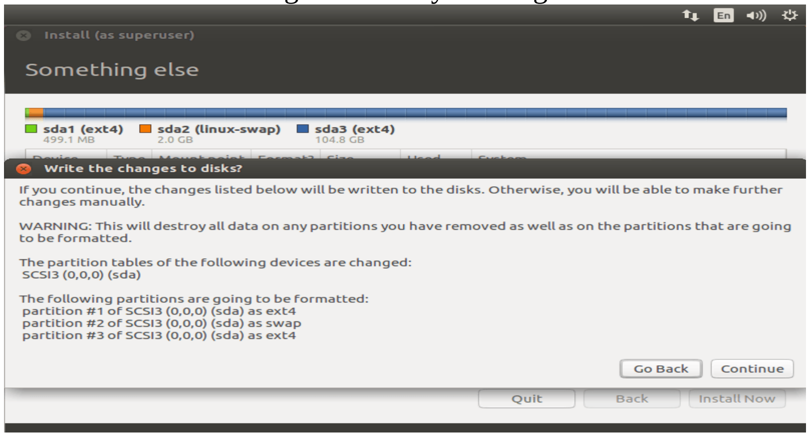
🟩 sda1 (ext4) 499.1 MB 🟧 sda2 (linux-swap) 2.0 GB 🟦 sda3 (ext4) 104.8 GB

Device	Type	Mount point	Format?	Size	Used	System
/dev/sda						
/dev/sda1	ext4	/boot	✓	499 MB	unknown	
/dev/sda2	swap			2047 MB	unknown	
/dev/sda3	ext4	/	✓	104825 MB	unknown	

Device for boot loader installation:

/dev/sda VMware, VMware Virtual S (107.4 GB

Install Ubuntu 16.04 – Partition List
Write the changes to disk by clicking on continue.


Install (as superuser)

Something else

🟩 sda1 (ext4) 499.1 MB 🟧 sda2 (linux-swap) 2.0 GB 🟦 sda3 (ext4) 104.8 GB

Write the changes to disks?

If you continue, the changes listed below will be written to the disks. Otherwise, you will be able to make further changes manually.

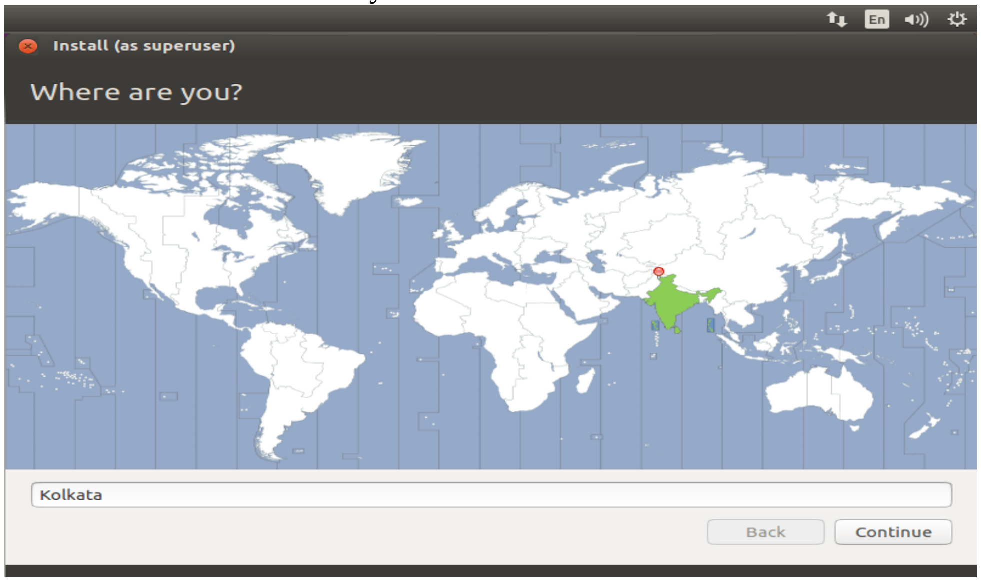
WARNING: This will destroy all data on any partitions you have removed as well as on the partitions that are going to be formatted.

The partition tables of the following devices are changed:
SCSI3 (0,0,0) (sda)

The following partitions are going to be formatted:
partition #1 of SCSI3 (0,0,0) (sda) as ext4
partition #2 of SCSI3 (0,0,0) (sda) as swap
partition #3 of SCSI3 (0,0,0) (sda) as ext4

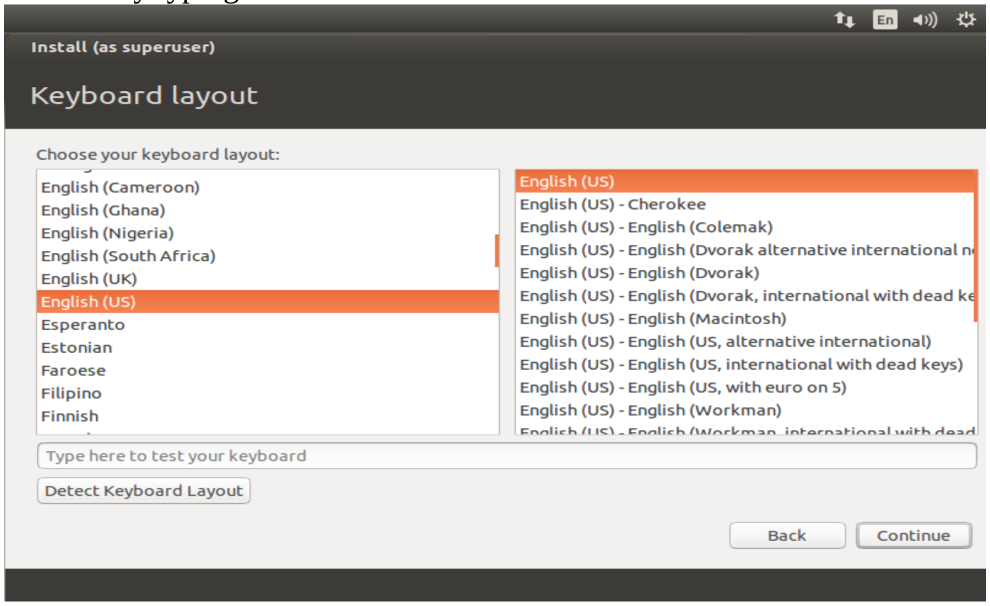
Install Ubuntu 16.04 – Formatting Partitions

Select your location next screen.



Install Ubuntu 16.04 – Select Location

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.



Install (as superuser)
Keyboard layout
Choose your keyboard layout:
English (Cameroon)
English (Ghana)
English (Nigeria)
English (South Africa)
English (UK)
English (US)
Esperanto
Estonian
Faroese
Filipino
Finnish
English (US)
English (US) - Cherokee
English (US) - English (Colemak)
English (US) - English (Dvorak alternative international n
English (US) - English (Dvorak)
English (US) - English (Dvorak, international with dead ke
English (US) - English (Macintosh)
English (US) - English (US, alternative international)
English (US) - English (US, international with dead keys)
English (US) - English (US, with euro on 5)
English (US) - English (Workman)
Type here to test your keyboard
Detect Keyboard Layout
Back
Continue

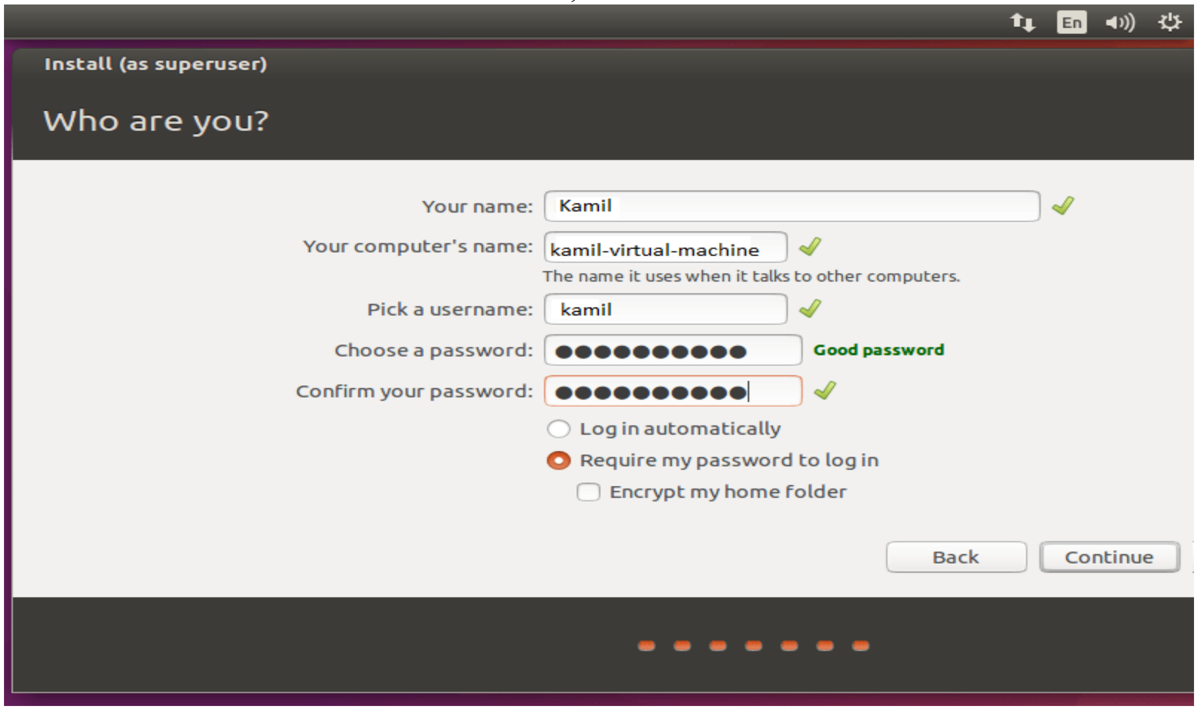
Ubuntu 16.04 – Keyboard Layout

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.



Install (as superuser)

Who are you?

Your name: Kamil ✓

Your computer's name: kamil-virtual-machine ✓

The name it uses when it talks to other computers.

Pick a username: kamil ✓

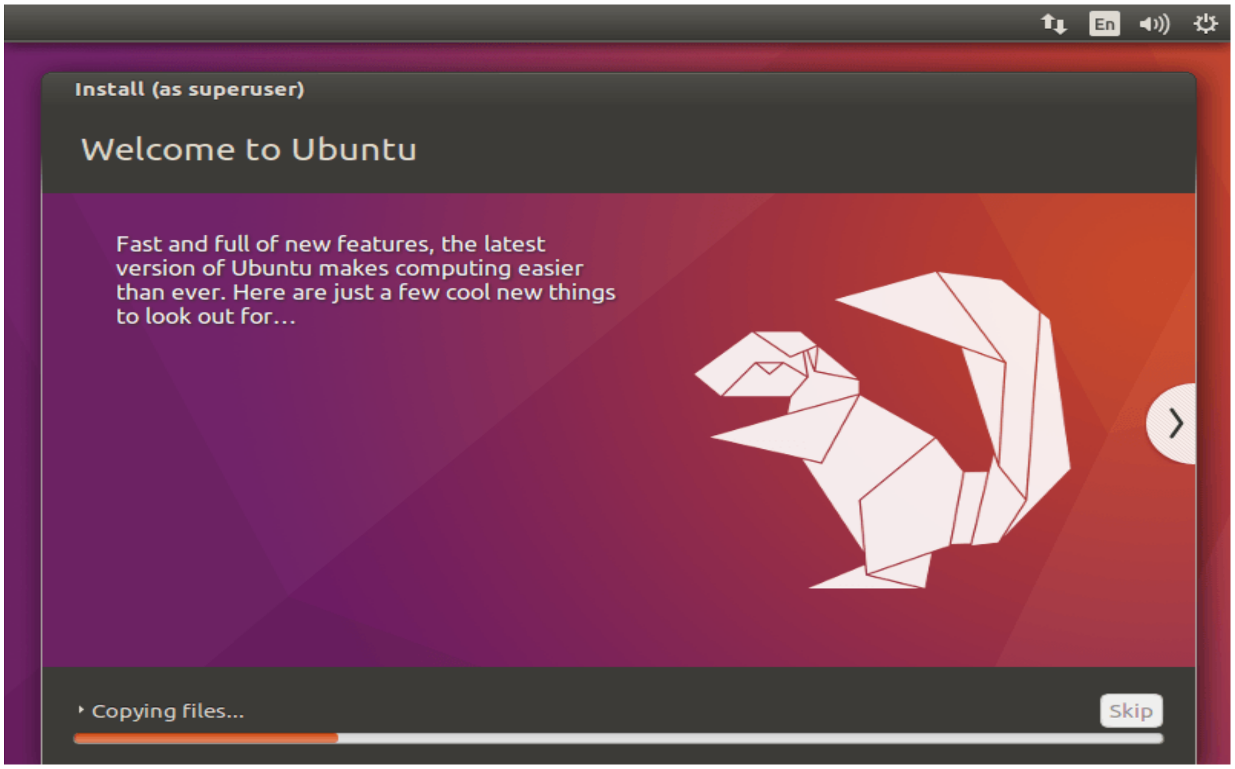
Choose a password: ●●●●●●●●●●● Good password

Confirm your password: ●●●●●●●●●●● ✓

Log in automatically [✓ Require my password to log in]

Encrypt my home folder

Install Ubuntu 16.04 – User



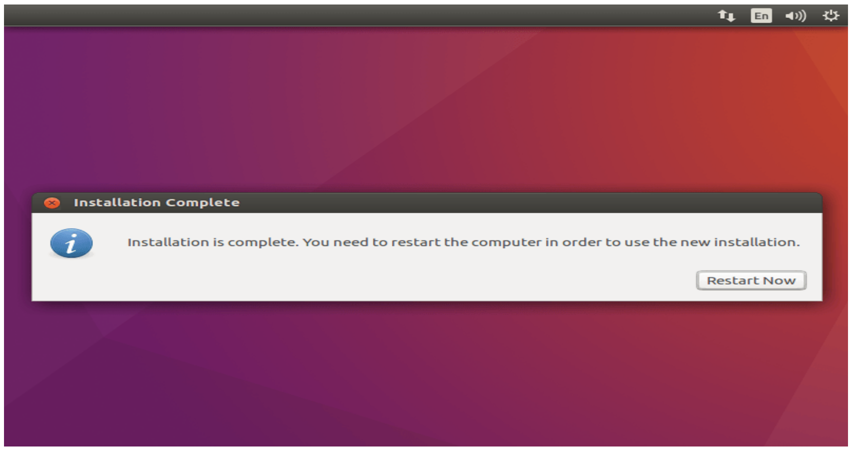
Install (as superuser)

Welcome to Ubuntu

Fast and full of new features, the latest version of Ubuntu makes computing easier than ever. Here are just a few cool new things to look out for…

Copying files...

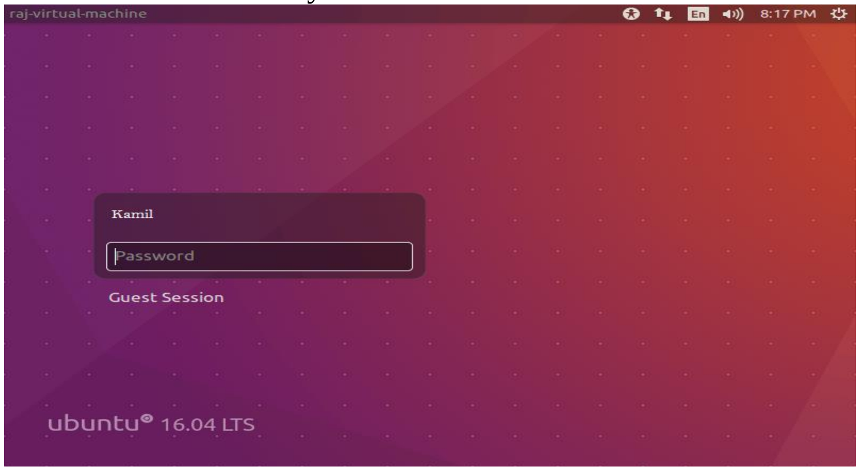
Below screenshot shows installing Ubuntu 16.04.
Install Ubuntu 16.04 – Installing
Once the installation is over, click on restart now.



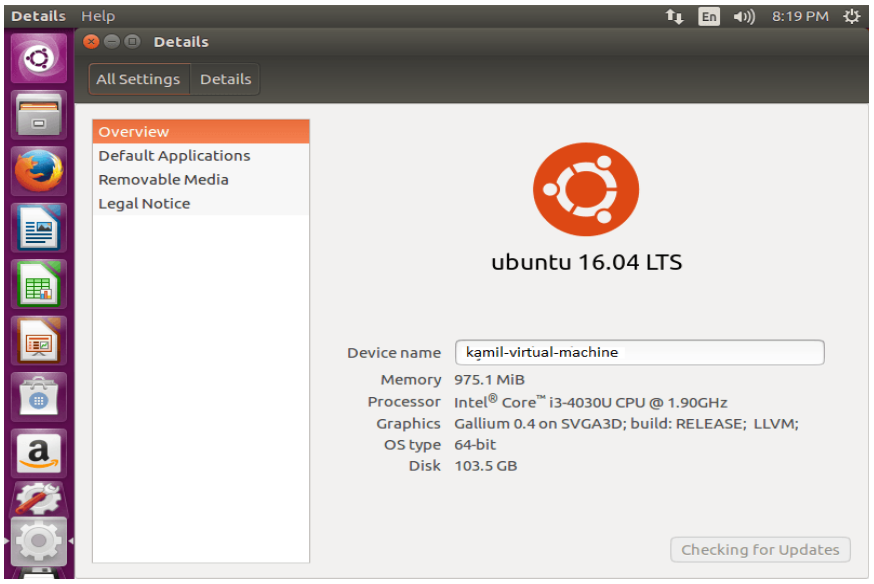
Installation is complete. You need to restart the computer in order to use the new installation.

Install Ubuntu 16.04 – Restart After the installation

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



Install Ubuntu 16.04 – Login Screen



Device name kamil-virtual-machine
Memory 975.1 MiB
Processor Intel® Core™ i3-4030U CPU @ 1.90GHz
Graphics Gallium 0.4 on SVGA3D; build: RELEASE; LLVM;
OS type 64-bit
Disk 103.5 GB

Install Ubuntu 16.04 – Desktop Screen

From a command shell, use the "concatenate" command: cat /proc/meminfo for memory usage information. You should see a line starting something like Mem: 64655360, etc. This is the total memory Linux thinks it has available to use.

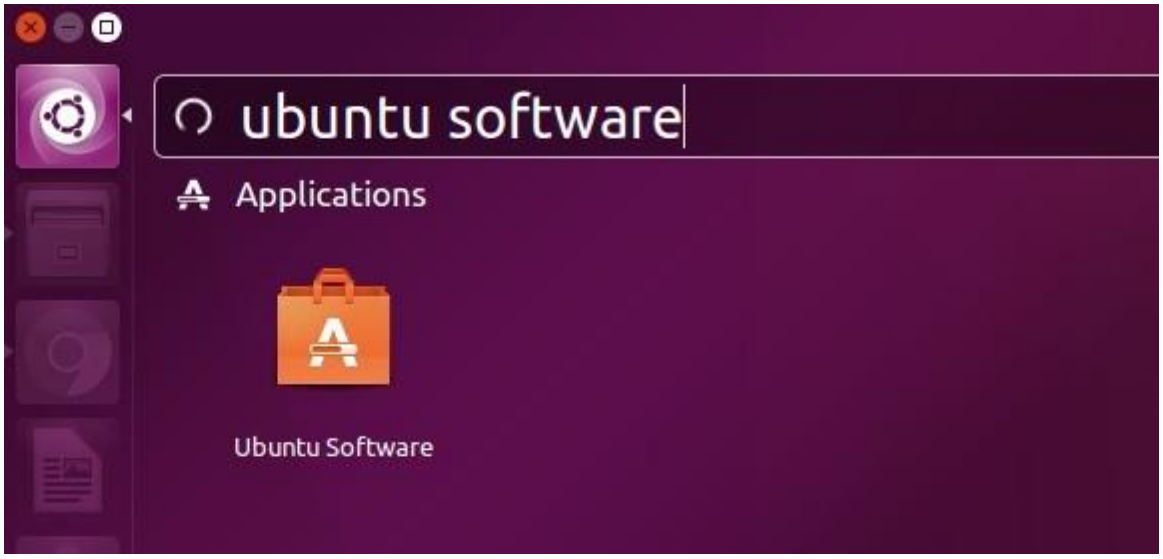
You can also use commands

- free - m
- vmstat
- top
- htop

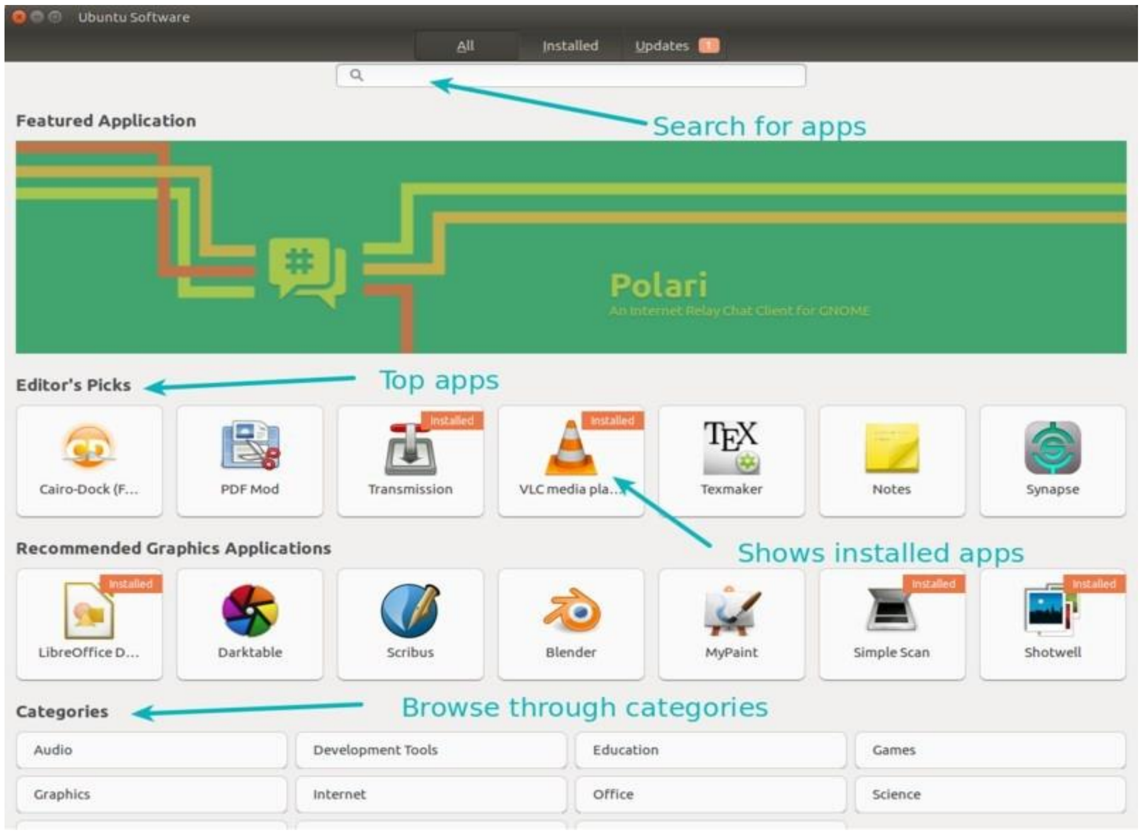
How to install applications in Ubuntu and how to remove installed software

1.1 Install software using Ubuntu Software Center
The easiest and most convenient way to find and install software in Ubuntu is by using Ubuntu Software Center. In Ubuntu Unity, you can search for Ubuntu Software Center in Dash and click on it to open it:

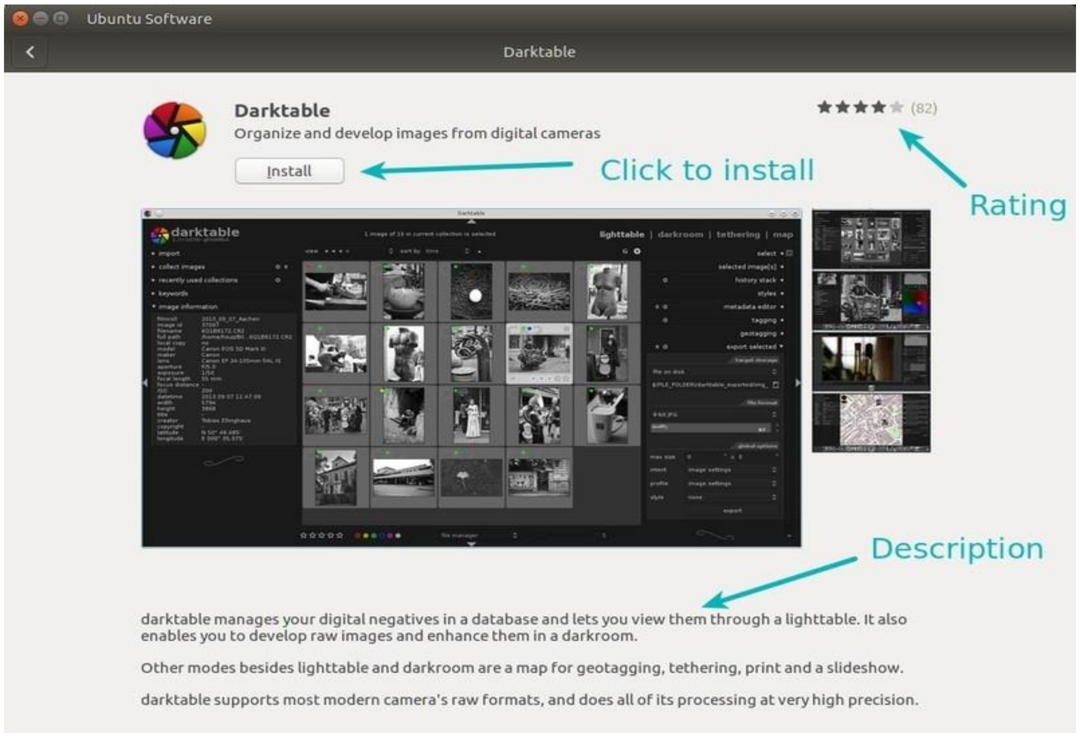
Run Ubuntu Software Center



You can think of Ubuntu Software Center as Google’s Play Store or Apple’s App Store. It showcases all the software available for your Ubuntu system. You can either search for an application by its name or just browse through various categories of software. You can also opt for the editor’s pick. Your choice mainly. Once you have found the application you are looking for, simply click on it. This will open a page inside Software Center with a description of the application. You can read the description, see its raiting and also read reviews. You can also write a review if you want.



Once you are convinced that you want the application, you can click on the install button to install the selected application. You'll have to enter your password in order to install applications in Ubuntu.



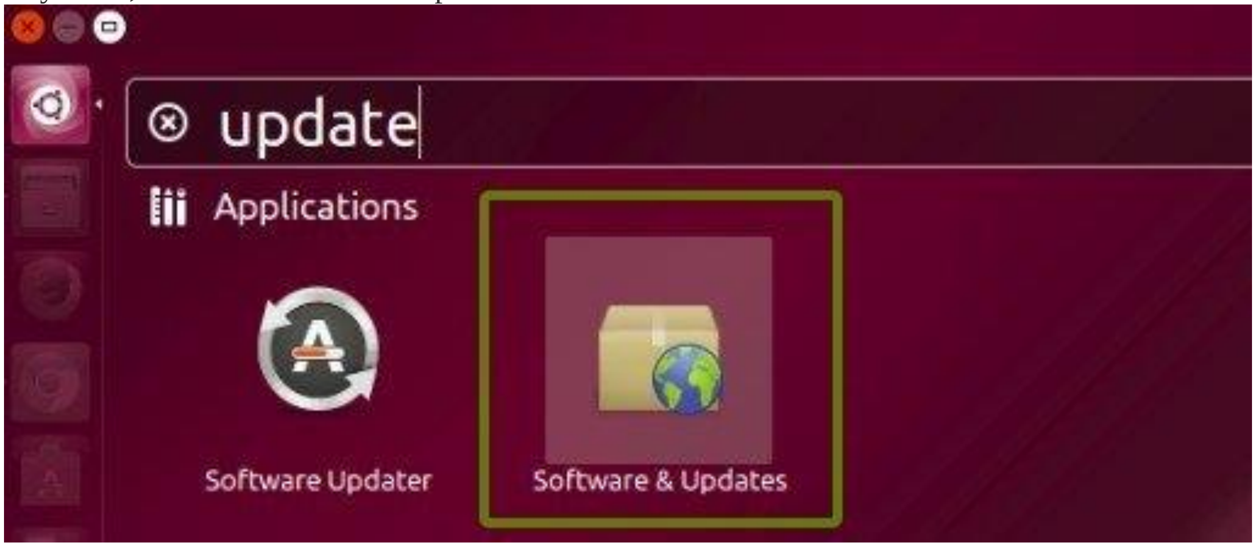
darktable manages your digital negatives in a database and lets you view them through a lighttable. It also enables you to develop raw images and enhance them in a darkroom.

Other modes besides lighttable and darkroom are a map for geotagging, tethering, print and a slideshow.

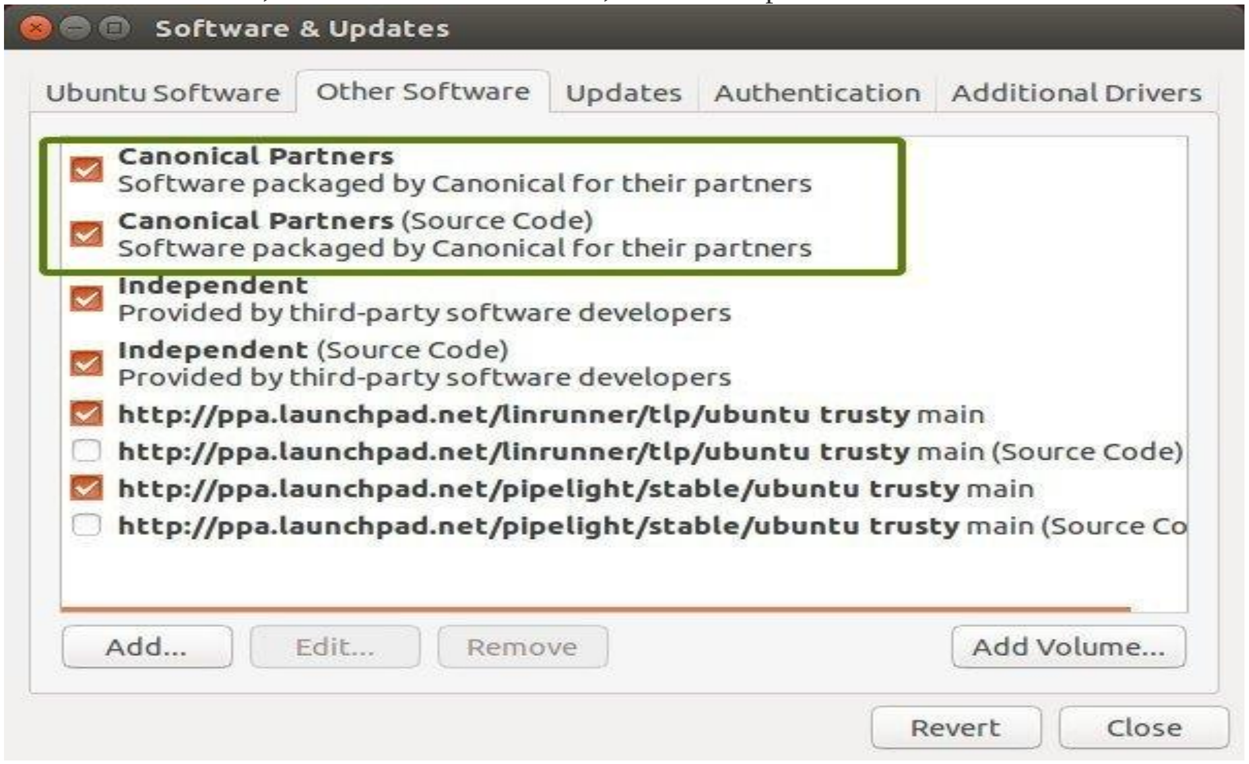
darktable supports most modern camera's raw formats, and does all of its processing at very high precision.

Check details and then install software

But there is also a Canonical Partner repository which is not directly controlled by Ubuntu and includes closed source proprietary software. Enabling this repository gives you access to more software. Installing Skype in Ubuntu is achieved by this method.
In Unity Dash, look for Software & Updates.



And in here, under Other Software tab, check the options of Canonical Partners.

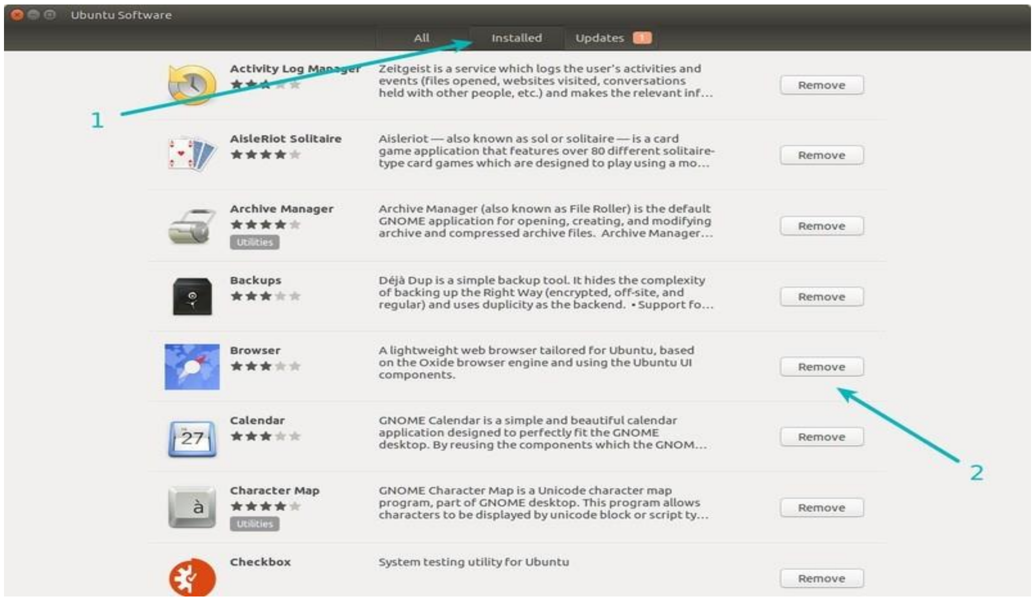

Software & Updates

Ubuntu Software | Other Software | Updates | Authentication | Additional Drivers

- [✓] **Canonical Partners**
  Software packaged by Canonical for their partners
- [✓] **Canonical Partners** (Source Code)
  Software packaged by Canonical for their partners
- [✓] **Independent**
  Provided by third-party software developers
- [✓] **Independent** (Source Code)
  Provided by third-party software developers
- [✓] **http://ppa.launchpad.net/linrunner/tlp/ubuntu trusty** main
- [ ] **http://ppa.launchpad.net/linrunner/tlp/ubuntu trusty** main (Source Code)
- [✓] **http://ppa.launchpad.net/pipelight/stable/ubuntu trusty** main
- [ ] **http://ppa.launchpad.net/pipelight/stable/ubuntu trusty** main (Source Co

Enable Canonical partners to access more software

1.2 Remove software using Ubuntu Software Center
We just saw how to install software using Ubuntu Software Center. How about removing software that you had installed using this method?
Uninstalling software with Ubuntu Software Center is as easy as the installation process itself.



Activity Log Manager — Zeitgeist is a service which logs the user's activities and events (files opened, websites visited, conversations held with other people, etc.) and makes the relevant inf... — Remove

AisleRiot Solitaire — Aisleriot — also known as sol or solitaire — is a card game application that features over 80 different solitaire-type card games which are designed to play using a mo... — Remove

Archive Manager (Utilities) — Archive Manager (also known as File Roller) is the default GNOME application for opening, creating, and modifying archive and compressed archive files. Archive Manager... — Remove

Backups — Déjà Dup is a simple backup tool. It hides the complexity of backing up the Right Way (encrypted, off-site, and regular) and uses duplicity as the backend. • Support fo... — Remove

Browser — A lightweight web browser tailored for Ubuntu, based on the Oxide browser engine and using the Ubuntu UI components. — Remove

Calendar — GNOME Calendar is a simple and beautiful calendar application designed to perfectly fit the GNOME desktop. By reusing the components which the GNOM... — Remove

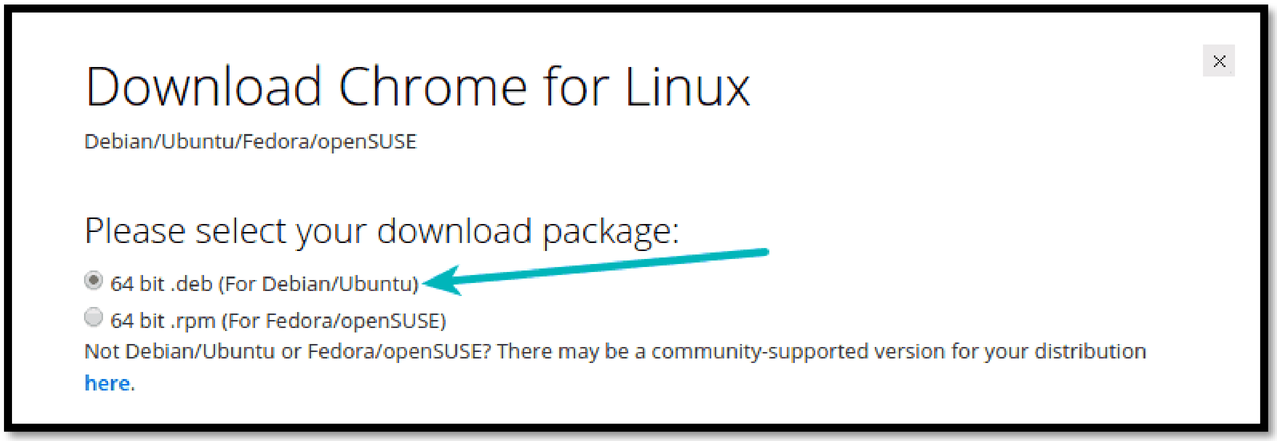
Character Map (Utilities) — GNOME Character Map is a Unicode character map program, part of GNOME desktop. This program allows characters to be displayed by unicode block or script ty... — Remove

Checkbox — System testing utility for Ubuntu — Remove

Open the Software Center and click on the Installed tab. It will show you all the installed software.

Alternatively, you can just search for the application by its name.
To remove the application from Ubuntu, simply click on Remove button. Again you will have to provide your password here.

1.3 Install software in Ubuntu using .deb files
.deb files are similar to the .exe files in Windows. This is an easy way to provide software installation. Many software vendors provide their software in .deb format. Google Chrome is such an example.
You can download .deb file from the official website.



Download Chrome for Linux

Debian/Ubuntu/Fedora/openSUSE

Please select your download package:

[✓ 64 bit .deb (For Debian/Ubuntu)]
64 bit .rpm (For Fedora/openSUSE)
Not Debian/Ubuntu or Fedora/openSUSE? There may be a community-supported version for your distribution here.

Once you have downloaded the .deb file, just double click on it to run it. It will open in Ubuntu Software Center and you can install it in the same way as we saw in section 1.1.
Alternatively, you can use a lightweight program Gdebi to install .deb files in Ubuntu.
Once you have installed the software, you are free to delete the downloaded .deb file.

- Make sure that you are downloading the .deb file from the official source. Only rely on the official website or GitHub pages.
- Make sure that you are downloading the .deb file for correct system type (32 bit or 64 bit). Read our quick guide to know if your Ubuntu system is 32 bit or 64 bit.

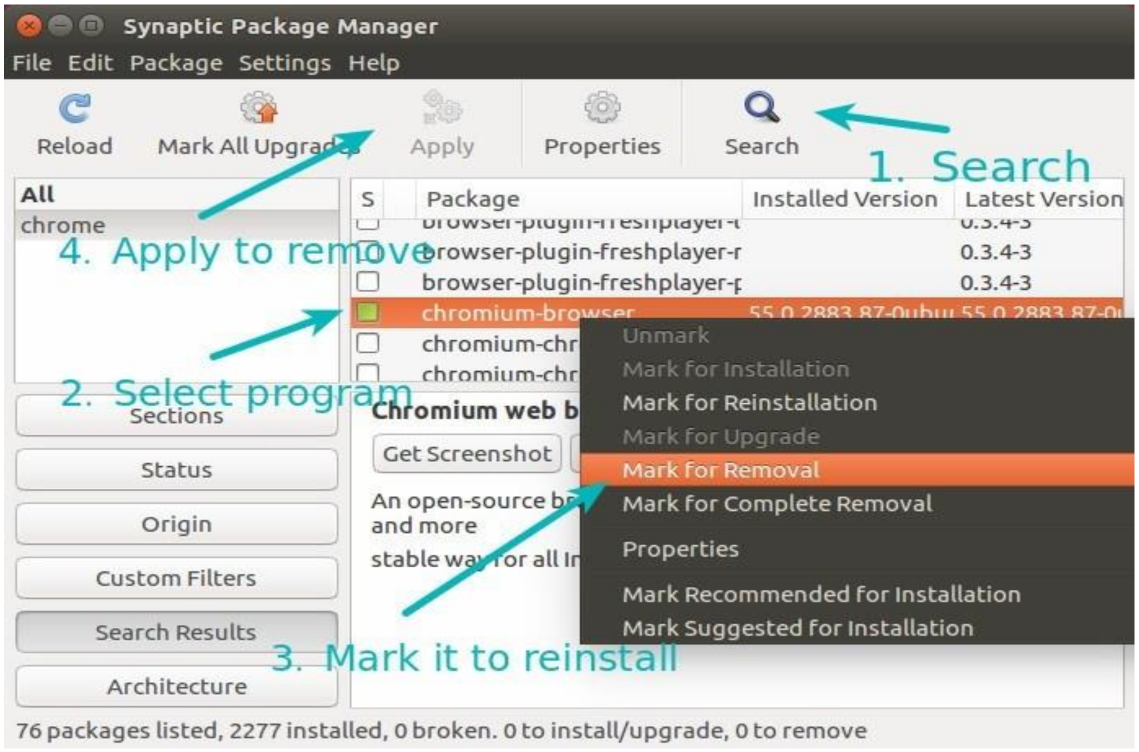
  1.4 Remove software that was installed using .deb

  Removing software that was installed by a .deb file is the same as we saw earlier in section 1.2. Just go to Ubuntu Software Center, search for the application name and click on remove to uninstall it.

  Alternatively, you can use Synaptic Package Manager. Not necessarily but this may happen that the installed application is not visible in Ubuntu Software Center. Synaptic Package Manager is lists all the software that are available for your system and all the software that are already installed on your system. This is a very powerful and very useful tool.

  This is a very powerful and very useful tool. Before Ubuntu Software Center came into existence to provide a more user-friendly approach to software installation, Synaptic was the default program for installing and uninstalling software in Ubuntu.

  You can install Synaptic package manager by clicking on the link below (it will open Ubuntu Software Center).



Open Synaptic Manager and then search for the software you want to uninstall. Installed softwares are marked with a green button. Click on it and select "mark for removal". Once you do that, click on "apply" to remove the selected software.

1.5 Install software in Ubuntu using apt commands [recommended]

You might have noticed a number of websites giving you a command like "sudo apt-get install" to install software in Ubuntu.

This is actually the command line equivalent of what we saw in section 1. Basically, instead of using the graphical interface of Ubuntu Software Center, you are using the command line interface. Nothing else changes.
Using the apt-get command to install software is extremely easy. All you need to do is to use a command like:

# -> sudo apt-get install package_name

Here sudo gives 'admin' or 'root' (in Linux term) privileges. You can replace package_name with the desired software name.
apt-get commands have auto-completion so if you type a few letters and hit tab, it will provide all the programs matching with those letters.
1.6 Remove software in Ubuntu using apt commands [recommended]
You can easily remove softwares that were installed using Ubuntu Software Center, apt command or .deb file using the command line.
All you have to do is to use the following command, just replace the package_name with the software name you want to delete.

# -> sudo apt-get remove package_name

Here again, you can benefit from auto completion by pressing the tab key.
Using apt-get commands is not rocket science. This is in fact very convenient. With these simple commands, you get acquainted with the command line part of Ubuntu Linux and it does help in long run. I recommend reading my detailed guide on using apt-get commands to learn in detail about it.

Install GCC

The following linux command will install `gcc` compiler on on Ubuntu 18.04 Bionic Beaver. Open up terminal and enter:

```
$ sudo apt install g++
```

Install build-essential

Another way to install `g++` compiler is to install it as part of `build-essential` package. Additionally the `build-essential` package will also install additional libraries as well as `gcc` compiler. In most cases or if unsure this is exactly what you need:

$ sudo apt install build-essential

Check G++ version

Confirm your installation by checking for GCC version:

```
$ g++ --version
g++ (Ubuntu 7.2.0-18ubuntu2) 7.2.0
Copyright (C) 2017 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

Same For GCC

SET A

1) **How can you find out how much memory Linux is using?**
2) **How do you switch from one desktop environment to another, such as switching from KDE to Gnome?**
3) **How do you execute more than one command or program from a single command line entry?**

SET B

1) **How can you install multiple packages using single command?**
2) **How to remove software that was installed using .deb?**
3) **Install all Python package.**

SET C

1) **Install the following Software using sudo apt install :**
   Before this use sudo apt update
   a) python3-pip
   b) gcc
   c) default-jdk
   d) libapache2-mod-php
   e) Apache2
   f) mysql-server

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well done [ ]

Signature of the instructor: ___________________________ Date: ________________

Assignment 2

Aim: To study of Desktop Environment and Components -
Pre-requisite: Knowledge of Linux.

Note: The student should read following topics before starting exercises.

Desktop environment (DE):

Desktop environment (DE) is an implementation of the desktop metaphor made of a bundle of programs running on top of a computer operating system, which share a common graphical user interface (GUI), sometimes described as a graphical shell.

The desktop environment was seen mostly on personal computers until the rise of mobile computing. Desktop GUIs help the user to easily access and edit files, while they usually do not provide access to all of the features found in the underlying operating system. Instead, the traditional command-line interface (CLI) is still used when full control over the operating system is required.

A desktop environment typically consists of icons, windows, toolbars, folders, wallpapers and desktop widgets (see Elements of graphical user interfaces and WIMP).

Best Desktop Environment for Linux:

Just to mention, this is not the list for the best desktop environment for programming or any such specific purposes. Also, the list is not in any specific order. There are lots of popular desktop environments available for Linux in the market but choosing the best one, depending on requirement, quite difficult for the newbie. So that we create this details on best Linux desktop environment. Details are as follows:

Sr. No.	Name of Desktop Environments	Pros	Cons
1.	KDE Plasma Desktop Environment	Most advanced, powerful and feature-rich, Modern and polished user interface, Highly customizable and flexible experience, Wide range of software, compatibility	Slightly resource heavy, Some components might seem too complicated to use
2.	MATE on Ubuntu MATE	Intuitive and robust experience, Simple and lightweight, Highly customizable	Interface might seem old (because it's intended to be)
3.	GNOME	Modern and touch-friendly UI Can extend functionalities through GNOME Shell Extensions , Customizable	Resource heavy, Extension management is not satisfactory
4.	Cinnamon on Linux Mint	Sleek and polished look, Familiar interface, Pretty customizable	Small icons are not touch friendly

5.	Budgie	Solid and intuitive Modern UI, elegant looks, Seamless desktop experience	Available only on few distributions (can be installed on your own manually)
6.	LXDE on Fedora	Extremely fast performing and lightweight, Supports almost every Linux distro	User interface may seem unappealing
7.	Unity	Intuitive, New technologies like HUD, Customizable by third-party applications	Poor implementation of notifications, Very little default tools for customization, Inconsistent UI

Desktop Components:

Instructor Notes: Highlight the fact that unlike other operating systems, Ubuntu comes with a completely clean desktop, by default. Users are free to add icons and files on the desktop according to their preferences.

GNOME is the default desktop environment for Ubuntu. GNOME (GNU Network Object Model Environment) is an international effort to build a complete desktop environment—the graphical user interface, which sits on top of a computer operating system—entirely from free software. This goal includes creating software development frameworks, selecting application software for the desktop and working on the programmes.

Which manage application launching, file handling and window and task management. Community members worldwide contribute to the translation and accessibility of the desktop in multiple languages.

Key Desktop Components on Ubuntu when you start your computer, the first screen displayed on Ubuntu is the logon screen, where you type your user name and password. The next screen displayed is the Ubuntu desktop. Ubuntu comes with a completely clean desktop background, free of icons by default.

Let's discuss various components like window manager, Panels, Menu, System Tray, Icons, Widgets, Launcher, dashboards, file manager, display manger, terminal emulator, text editor, configuration tools etc.

1) **Window manager:**

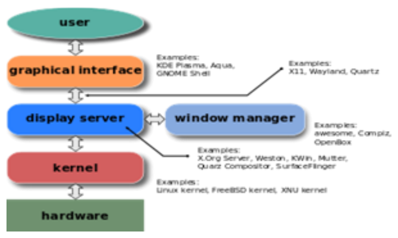

user
graphical interface
Examples: KDE Plasma, Aqua, GNOME Shell
Examples: X11, Wayland, Quartz
display server
window manager
Examples: awesome, Compiz, OpenBox
Examples: X.Org Server, Weston, KWin, Mutter, Quarz Compositor, SurfaceFlinger
kernel
Examples: Linux kernel, FreeBSD kernel, XNU kernel
hardware

Figure 1

An X window manager is a window manager which runs on top of the X Window System, a windowing system mainly used on Unix-like systems. Unlike the classic Mac OS, macOS (Apple Macintosh) and Microsoft Windows platforms (excepting Microsoft Windows explorer.exe shell replacements) which have historically provided a vendor-controlled, fixed set of ways to control how windows and panes display on a screen, and how the user may interact with them, window management for the X Window System was deliberately kept separate from the software providing the graphical display. The user can choose between various third-party window managers, which differ from one another in several ways, including:

- customizability of appearance and functionality:
  - textual menus used to start programs and/or change options
  - docks and other graphical ways to start programs
  - multiple desktops and virtual desktops (desktops larger than the physical monitor size), and pagers[1] to switch between them
- consumption of memory and other system resources
- Degree of integration with a desktop environment, which provides a more complete interface to the operating system, and provides a range of integrated utilities and applications.

2) Gnome Panel: Panel is a component that is part of Gnome Flashback and provides **panels** and default applets for the desktop. A **panel** is a horizontal or vertical **bar** that can be added to each side of the screen. On the bottom **panel** there is usually a list of open applications. Gnome Panel component comes with several default applets such as

- The Workspace Switcher
- The Window List
- The Window Selector
- The Notification Area
- The Clock
- And the infamous 'Wanda the Fish'

Panel Configuration:
To configure a panel you can hold the <ALT>-key and right click on the panel, this will open a context menu that allows you to:

- Add an applet
- Configure the panel through "Properties"
- Delete the current panel
- Or add a new panel

3) How to Enable Local Menus in Ubuntu

There's no "**System**" **menu** in modern versions of **Ubuntu**. Just open the Dash (using **Ubuntu** button on the Launcher or Win key on your keyboard) and start typing program's name that you want to launch.

1. The Global menu is a feature in Ubuntu that places the menu bar for each application on the bar at the top of the screen. ...
2. On the System Settings dialog box, click the “Appearance” icon in the Personal section.

3. On the Appearance screen, click the “Behavior” tab.

4) System Tray:
System Places menu, you will find the system tray. The system tray holds buttons for commonly used applications. On Ubuntu these include icons for the Konqueror Web Browser, Amarok media player, Kopete Instant Messenger, and KContact contact address book.

5) Icons: While Ubuntu comes with a number of icon packs you can select from, online a small number is installed by default. The first place to look for more icons is on the default repositories.

6) Widgets:
Screen less is a framework that allows adding widgets to your desktop. **The Screen less PPA provides numerous screen less (desktop widgets), such as RSS readers, weather, clock, countdown, a Conky-like system information widget, folder view, calendars, sensors, and much more.**
7) Launcher: Unity **Launchers** are actually files stored in your computer, with a '.desktop' extension. In earlier **Ubuntu** versions, these files were simply used so as to launch a specific application, but in Unity they are also used so as to create right-click menus for each application, which you can access from the Unity **Launcher.** How do I get desktop launcher on Ubuntu?

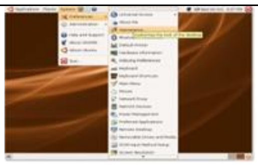
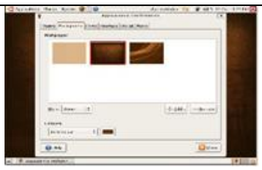
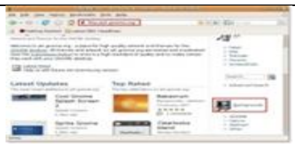
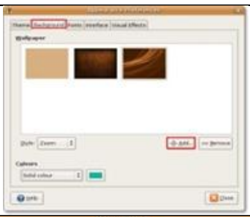
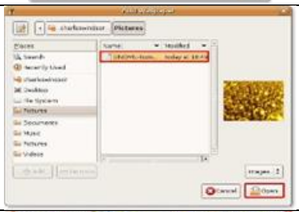
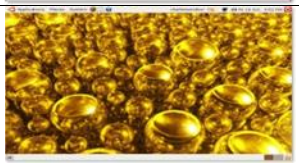
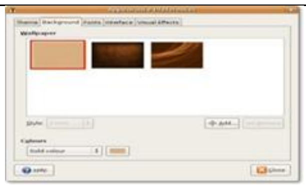
1. Right-click unused space in any panel (the toolbars at the top and/or bottom of the screen)
2. Choose Add To Panel...
3. Choose Custom Application Launcher.
4. Fill in Name, Command, and Comment. ...
5. Click the No Icon button to select an icon for your launcher. ...
6. Click OK.
7. Your launcher should now appear on the panel.

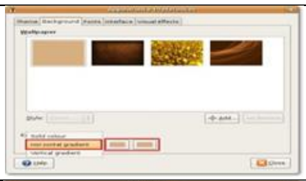
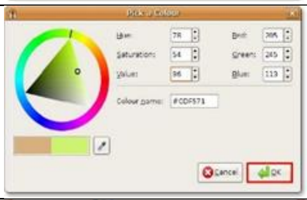
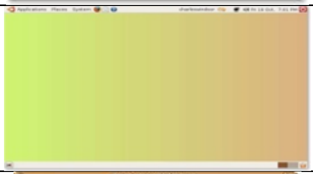
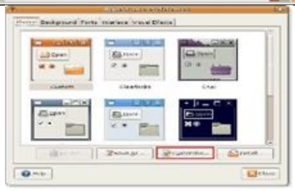
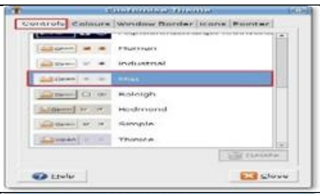
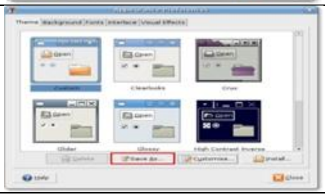
8) Terminal Emulator: Guake, ROXTerm, XTerm, Eterm, Gnome Terminal are number of terminal emulators.

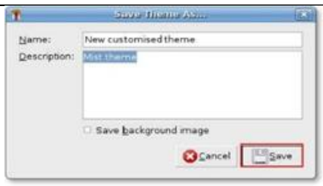
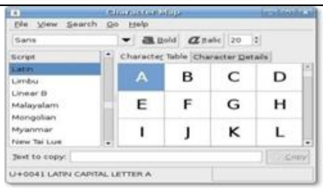
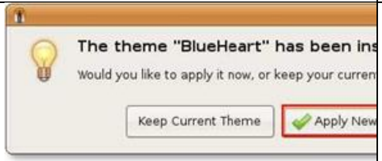
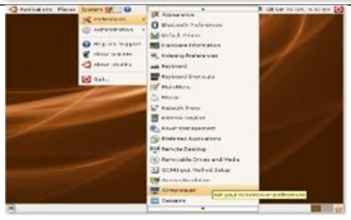
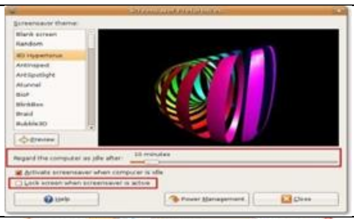
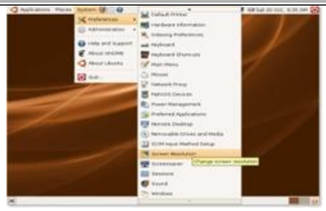
- Sakura. Sakura is a terminal emulator based just on GTK and VTE.
- LilyTerm. LilyTerm is a terminal emulator based off of libvte that aims to be fast and lightweight, Licensed under GPLv3.
- Konsole. If you're a KDE or Plasma user, you must know Konsole.

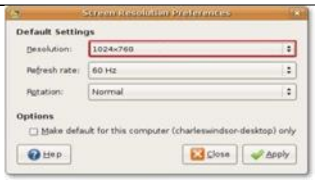
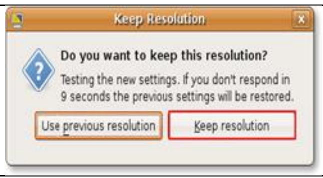
Customizing the Desktop Ubuntu and its derivatives can be customized through a Graphical User Interface or a Command Line Interface. The graphical tools to customize the desktop are available as menu options in the System menu. Point to Preference s on the System menu to view the tools.

Changing the Background: The desktop background is the image or color applied to your desktop.

Screen	Description
	Figure 2: Launching Appearance Preferences Dialogue Box On the System menu, point to Preferences and then click Appearance. The Appearance Preferences dialogue box opens. **Nice to Know:** You can also right-click the desktop and select Change Desktop Background to open the Appearance Preferences dialogue box.
	Figure 3: Changing the Desktop Wallpaper The background changes immediately. **Nice to Know:** To view the name of the wallpaper, move the pointer over its name.
	Figure 4: Applying Preference Change Click Close in the Appearance Preferences dialogue box to apply the changes. In addition to the wallpapers available with Ubuntu, you can download wallpapers from other sources and add them to the available wallpapers list in the Appearance Preferences dialogue box.
	Figure 5: Opening Wallpaper Source 1. Open the Web site http://art.gnome.org/ and click Backgrounds. 2. Download the wallpaper of your choice. 3. On the System menu, point to Preferences and then click Appearance. The Appearance Preferences dialogue box opens.
	Figure 6:Adding a New Wallpaper In the Add Wallpaper dialogue box, select the downloaded image and click Open.
	Figure 7: Selecting Downloaded Wallpaper This step adds the image as new wallpaper. •Click Close in the Appearance Preferences dialogue box to accept the changes. You can now view the new desktop background.
	Figure 8: Added Wallpaper Nice to know: You can, of course, use a picture from any other source to use as your desktop background. Many popular online photo collaboration sites allow visitors to download and use their content for personal use. Many people also use their own digital photographs as backgrounds.
	**To change the color of the background:** Figure 9: Changing Background Color On the System menu, point to Preferences and then click Appearance to open the Appearance Preferences dialogue box.

	Click the Background tab and select the wallpaper No Wallpaper. You can only view colors if you have not set any desktop wallpaper. The Colors box provides three types of background: Solid color, Horizontal gradient and Vertical gradient. Select the desktop color of your choice and then click the color chip next to the Colors box. The Pick a Colour dialogue box opens.
	Figure 10**:** Selecting a colour Option Select a colour or the attributes of a colour such as hue and saturation to create a colour of your choice. Click OK. The desktop reflects the new settings immediately.
 Hue: 78 Saturation: 54 Value: 96 Red: 205 Green: 245 Blue: 113 Colour name: #CDF571	Figure 11: Specifying Colour Click Close to close the Appearance Preferences dialogue box.
	Figure 12: Changed Background Colour
	**Customizing the Theme (Buttons & icon set c)** The desktop theme controls the visual appearance of the buttons, scroll bars, icons, panels, borders etc. A number of themes are provided with Ubuntu. On the System menu, point to Preferences and click Appearance. The Appearance Preferences dialogue box opens. On the Theme tab, select the theme of your choice. The desktop reflects the theme automatically. To customize your theme further, click Customize. The Customize Theme dialogue box opens
	Figure 13: Customizing Desktop Theme • The default selection is Controls tab. The setting on the Controls tabbed page defines the visual appearance of windows, panels and applets. Select a control from the Controls list. You will see an immediate change in the appearance of the open windows.
	Figure 14: Selecting Theme Controls ClickCloseintheCustomiseThemedialoguebox.Tosavethetheme,clickSaveAsintheAppearancePreferences dialogue box. The Save Theme As dialogue box opens.

 Save Theme As... Name: New customised theme Description: Mist theme	Figure 15: Saving a Modified Theme Provide a name for the theme in the Name box and a description, if you want, in the Description box. Click Save.
 Character Map File View Search Go Help Script Latin Limbu Linear B Malayalam Mongolian Myanmar New Tai Lue Character Table Character Details A B C D E F G H I J K L Text to copy: Copy U+0041 LATIN CAPITAL LETTER A	Figure 16: Specifying Theme Name and Description In the Appearance Preferences dialogue box, click Close. If you open a menu or window, you can see the changes in their appearance.
	Figure 17: Selecting the Downloaded Theme You can apply a new theme or retain the current theme. Click Apply New Theme to apply the new theme. The screen will reflect the new theme immediately.
 The theme "BlueHeart" has been ins Would you like to apply it now, or keep your curren	Figure 18: Applying New Theme Click Close in the Appearance Preferences dialogue box. If you open any menu or window, it will reflect the selected theme.
	Figure 19: Launching Screensaver Preferences Dialogue Box A screensaver displays (often moving) images on the screen when your computer is switched on but not in use. To go back to the workspace, you can move the mouse or press any key on the keyboard. On the System menu, point to Preferences and click Screensaver. The Screensaver Preferences dialogue box opens. Select a screensaver from the list. You can preview the screensaver in the right pane.
	Figure 24: Customising Screensaver Settings The Regard the computer as idle after slider specifies when a screensaver starts to work if the computer is not in use. The default time is set to 10 minutes. You can use this slider to select how long the computer needs to be idle before the screensaver activates.
	Figure 25: Launching Screen Resolution Preferences Dialogue Box **Customizing the Screen Resolution** The screen resolution determines how large or small an item looks on the screen. On the System menu, point to Preferences and then click Screen Resolution. The Screen Resolution Preferences dialogue box opens. The default resolution is 1280x1024. You can change the resolution in the Resolution box.

 Screen Resolution Preferences **Default Settings** Resolution: 1024x768 Refresh rate: 60 Hz Rotation: Normal **Options** ☐ Make default for this computer (charleswindsor-desktop) only Help Close Apply	Figure 26: Customising Screen Resolution Click Apply. The Keep Resolution dialogue box opens, prompting you to confirm settings or use the previous resolution and revert to the original settings. Click Keep resolution to apply new changes.
 Keep Resolution **Do you want to keep this resolution?** Testing the new settings. If you don't respond in 9 seconds the previous settings will be restored. Use previous resolution Keep resolution	Figure 27: Resolution Confirmation Dialogue Box The screen resolution will change.

Exercises:

Set A:

1) What is the default desktop for Ubuntu?
2) Enlist different Desktop environment. Also choose one for your operating system and apply it.
3) Configure panel for your system.
4) Enable Local Menus in Ubuntu.
5) Launch desktop launcher on Ubuntu

Set B:

1) Change your desktop *Background and wallpaper*.
2) Select color box and Select the desktop color of your choice and apply color to your background.
3) On the System menu, point to Preferences and click Screensaver. The Screensaver Preferences dialogue box opens. Select screensaver list and apply any one you like for your system.
4) Customize your window borders and icons.
5) Specifying Theme Name and Description for your system and View an Application in a Modified Theme

Set C:

1. Launch a Screen Resolution Preferences Dialogue Box and change default resolution by using Customize Screen Resolution.
2. Change the time and date of your system.

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well done [ ]

Signature of the instructor: ________________ **Date:** _________________

Assignment -3

Aim: - To study different basic Linux Commands.
Pre-requisite: Knowledge of Linux operating system environment.

Students should read following topics before solving exercise.
LINUX system will usually offer a variety of shell types:

- sh or Bourne Shell: the original shell still used on UNIX systems and in UNIX-related environments. It is available on every Linux system for compatibility with UNIX programs.
- bash or Bourne Again shell: the standard GNU shell, is the standard shell for common users on Linux and is a super set of the Bourne shell.
- csh or C shell: the syntax of this shell resembles that of the C programming language.
- tcsh or Turbo C shell: a superset of the common C shell, enhancing user-friendliness and speed.
- ksh or the Korn shell: A superset of the Bourne shell

All LINUX commands are case sensitive single words optionally having arguments. One of the argument is options which starts with "–"sign immediately followed by one or more characters indicating option. The wild-cards or Meta characters "*" and "?" Have similar meaning as in DOS. The "*" character matches any number of characters while"?" matches a single character. The backquote " ` " is another metacharacter. Shell executes the command enclosed in backquote in its place. Any wild-card is escaped with a \ character to be treated as it is.

Terminal Control Characters

Character	**Meaning**
^**h**	Backspaces one column.
^**u**	**Ctrl U** erases everything from the current cursor position to the beginning of the line.
^**d**	**Ctrl D** tells the **terminal** that it should register a EOF on standard input, which bash interprets as a desire to exit. **Ctrl D** means end of file.
^**c**	**Ctrl C** is used to kill a process with the signal SIGINT , and can be intercepted by a program so it can clean its self up before exiting, or not exit at all.
^**s**	**Ctrl S**. Stop all output to the screen. This is particularly useful when running commands with a lot of long, verbose output,
^**q**	**Ctrl** Q Resume output to the screen after stopping it with **Ctrl S**.
^**z**	**Ctrl Z** is used for suspending a process by sending it the signal SIGSTOP , which cannot be intercepted by the program.

Command	Meaning	Example
bc	A calculator that reads from standard input	E.g $echo "100 / 3 "| bc
man	Show documentation for a command	 $man cat
printf	Format and print data	E.g. printf 'My name is %s' 'Ashwini'
date	Print or parse date strings	To display Date date Display the date by parsing a given string with date –d Eg. date –d '2013-01-03'
echo	Print arguments to standard output	To Print "something" to screen $echo "something"
whoami	Print your username.	E.g. who am i
cal	Displays calender of current month	E.g $cal
tty	Displays Current terminal	E.g.$tty
info	Info documents are sometimes more elaborate than the man pages. These are like web pages. Internal links are present within the info pages.	E.g. $ info date

uname	Displays the information about the system.	1) $uname[option]
		E.g.
		$ uname –a
		2) It prints all the system information.
		$ uname –s
		3) It prints Kernel Name.
		$ uname –n
		4) It prints hostname of network node.

stty	It is used to change and print terminal line settings.	E.g. • **$ stty –all** This option print all current settings in human-readable form. • **$stty–g** This option will print all current settings in a stty-readable form. • **$stty --help** This option will display this help and exit.
script	**script** command in Linux is used to	script [options] [file]

	make typescript or record all the terminal activities. After executing the *script* command it starts recording everything printed on the screen including the inputs and outputs until exit.	E.g.
		$ script information.txt
		The output will contain content of Information.txt file,created by script command.
		-c
		command
		This option is used when we want to run a particular command.
		$ script –c cal calender.txt
		-e
		–return
		This option simply return exit code of the child process.
		-f
		–flush
		This option is used to run flush output after each write.

tput	Using tput you can control the color and cursor of your terminal.	• Set the Cursor Position using tput cup. E.g.$ tput cup 2 3 • Clear the Screen Using tput clear E.g.$tput clear • Get the Number of Columns and Lines of a Terminal E.g. $ tput cols $tput lines • Change the Terminal Background Color using tputsetb E.g. $tputsetb 4 • Change the Foreground Color using tputsetf E.g. $tputsetf 4
exit	**exit** command in linux is used to exit the shell where it is currently running.	E.g. $ exit Exit command with parameter $ exit 100 After pressing enter, the terminal window will close and return a status of 100. Return status is important because sometimes they can be mapped to tell error, warnings and notifications.

General Purpose Commands

Understanding environmental variables

Environment variables are dynamic values which affect the processes or programs on a computer. They exist in every operating system, but types may vary. Environment variables can be created, edited, saved, and deleted and give information about the system behavior.

Environment variables can change the way a software/programs behave.

Command	**Meaning**	**Example**
home	Gives home path for commands	$ HOME
path	Gives search path for commands	$ PATH
user	Gives current user's name	$ USER
uid	Gives user id of current user.	$ UID
editor	Gives deafault file editor.	$ EDITOR
term	Default terminal emulator. The TERM environment variable is used for terminal handling	$TERM
ps1	The command prompt and terminal appearance are governed by an environment variable called ps1. **PS1** represents the primary prompt string which is displayed when the shell is ready to read a command.	$ PS1
ps2	Continuation interactive prompt	$ PS2

Files and Directory Handling Commands

Linux Files and directories

Linux defines three main types of files. Linux treats all devices also as files.

- Ordinary or regular file -- A file containing data or program
- Directory file-- A file containing the list of filenames and their unique identifiers
- Special or device file --A file assigned to a device attached to a system

Linux files may or may not have extensions. A file can have any number of dots in its name. Linux file names are case sensitive. The root directory represented by / is the topmost directory file containing number of subdirectories which in turn contains subdirectories and files.

Directory Handling Commands

Command	**Meaning**	**Example**
pwd	This command displays the present working directory where you are currently in.	$ pwd
mkdir	This command will create a new directory, provided it doesn't exists.	$ mkdir FYBCA

cd	This command is used to change directory.	$ cd /
rmdir	This command will remove/delete an existing directory, provided it is empty.	$ rmdir FYBCA
cd ..	This command will take us one level up the directory tree.	$ cd ..

File Handling Commands in Linux

Command	**Meaning**	**Example**

touch	It is used to create a file without any content. The file created using touch command is empty. This command can be used when the user doesn't have data to store at the time of file creation.	1) Command to create file.
		$ touch File_name
		2) Command to create multiple files.
		$ touch File_name1 File_name2 File_name3
		3) To Change access time of file
		$ touch –a
		4) To change the modification time only.
		$ touch -m.

cat	Cat command reads data from the file and gives their content as output. It helps us to create, view, concatenate files.	1) To read a single file.
		$ cat filename
		2) **To view multiple files**
		$ cat filename1 filename2
		3) **To view contents of a file preceding with line numbers**
		$ cat-n filename
		4) **Cat command can append the contents of one file to the end of another file.**
		$ cat file1 >>file2
		5) **Cat command can display content in reverse order using tac command.**
		$ tac filename

cp	This command is used to copy files or group of files or directory	$ cp [option] source destination
		$ cp [option] source directory
		$ cp [option] source-1 source-2 source-n directory

ls	**ls** is a Linux shell command that lists directory contents of files and directories.	1) List files with ls with no option. $ ls 2) **Open Last Edited File** **$ls -t** 3) **Display All Information About Files/Directories** **$ls –l** 4) **isplay File Size in Human Readable Format** **$ls –lh** 5) **Order Files Based on Last Modified Time** **$ls –lt** 6) **Display Hidden Files** **$ls -a**
mv	mv is used to move one or more files or directories from one place to another in file system	mv [option]source destination E.g. $ mv a.txt geek.txt
rm	rm command is used to remove objects such as files, directories, symbolic links and so on from the file system	rm [option] …file… E.g. $ rm a.txt b.txt
file	**file command** is used to determine the type of a file.	file [option] [filename]
head	The head command, as the name implies, print the top N number of data of the given input. By default, it prints the first 10 lines of the specified files.	head [option] …file… E.g. $ head state.txt
tail	The tail command, as the name implies, print the last N number of data of the given input. By default it prints the last 10 lines of the specified files.	tail [option] …file… E.g. $ tail state.txt
more	**more** command is used to view the text files in the command prompt, displaying one screen at a time in case the file is large	more [-options] [num][+/pattern][+linenum] [filename]
less	less command is linux utility which can be used to read contents of text file one page(one screen) per time.	less filename
find	It can be used to find files and directories and perform subsequent operations on them.	$ find [where to start searching from] [expression determines what to find] [-options] [what to find]

diff	diff stands for **difference**. This command is used to display the differences in the files by comparing the files line by line.	diff [option] file1 file2
cmp	**cmp** command in Linux/UNIX is used to compare the two files byte by byte and helps you to find out whether the two files are identical or not.	$cmp [OPTION]... FILE1 [FILE2 [SKIP1 [SKIP2]]]
comm	comm compare two sorted files line by line and write to standard output the lines that are common and the lines that are unique.	$comm[option]….file1 file2
tar	The Linux ‘tar’ stands for tape archive, is used to create Archive and extract the Archive files.	$ tar [options] [archive file] [file or directory to be archived]
zip	zip is used to compress the files to reduce file size and also used as file package utility.	zip [option] zipfilefiles_list E.g. $ myfile.zip filename.txt
unzip	unzip will list, test, or extract files from a ZIP archive	unzip myfile.zip
passwd	*passwd* command in Linux is used to change the user account passwords.	$ passwd
patch	Patch is a command that is used to apply patch files to the files like source code, configuration. Patch files holds the difference between original file and new file.	patch [options] [original filename] [patchfile]]

Exercises:

Set A:

1) Explore all the UNIX commands given in this manual.
2) Create a directory.
3) Create a subdirectory in the directory created.
4) Change your current directory to the subdirectory.
5) Display the calendar for the current month.
6) Get a directory listing of the parent directory.
7) How many users were logged onto your system?
8) Display your name in the form of a banner.
9) Display the name of device name of your terminal.
10) Move to the root directory.

Set B:

1. Create a directory named FYBCA under that create a 3 directories CO, RDBMS and O.S. Create files under subdirectories CO and RDBMS and move these files to O.S.
2. Display username, user id, hostname, kernel name along with system information.
4. Display list of all files ending with .txt from current working directory.
5. Copy the file MyFile.txt to directory assignment1 and rename it to t_1.txt.
6. Display last modification and access time of particular file.
7. Create file student.txt and display size of file in bytes.
8. Display first and last 5 lines of student.txt
9. Prepare two text files and check output of diff, comm and cmp commands.

Set C:

1. Make a list of all filenames in /etc that contain the string samba.
2. Make a sorted list of all files in /etc that contain the case insensitive string samba.
3. Write a line that receives cities.txt file and sort that file.
4. Accept text file and display second to the seventh character of that file.
5. Create a text file which contains name of cities include city name pune in that file Display Pune along with previous and next city name.
6. Put a sorted list of all logged on users in onlineusers.txt.
7. Accept the file and display that file along with line numbers.

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well done [ ]

Signature of the instructor: ___________________ Date: _____________________

Assignment 4

Aim: To study redirection and simple filters in Linux commands.
Pre-requisite: Knowledge of Basic Linux Commands.

The student should read following topics before starting exercise.

Redirection and pipes

Redirection is a feature in Linux such that when executing a command, you can change the standard input/output devices. The basic workflow of any Linux command is that it takes an input and give an output.

With redirection, the above standard input/output can be changed. The default files where a command reads its input, sends its output and error messages are called standard input(stdin), standard output(stdout) and standard error(stderr) respectively.

By default all the above three files are attached with the terminal on which the command is executing. Therefore, every command, by default, takes its input from the keyboard and sends its output and error messages to the display screen. Redirection is used to detach default file from the command and attach some specific file.

Pipes allow you to send output of one command as input to the other command. The commands that are connected via a pipe are called filters.

In addition to redirecting input/output to a named file, you can connect two commands together so that the output from one program becomes the input of the next program. Two or more commands connected in this way form a pipe. To make a pipe, put a vertical bar (|) on the command line between two commands. When a pipe is set up between two commands, the standard output of the command to the left of the pipe symbol becomes the standard input of the command to the right of the pipe symbol.

Command	**Symbol**	**Description Format &**	**Example**
Input Redirection	<	Redirection It detaches the keyboard from the standard input of command and attaches specific file	$cat < abc.txt Takes its input from abc.txt and the output by default is on console. The effect is same as $cat tempfile
Output Redirection	>	It detaches the console from the standard output of command and attaches specific file	$cat > file1 Takes its input from keyboard by default and writes the output to file1, effectively whatever typed at the keyboard goes into tempfile $cat file1 abc.txt >

			file2 The contents of file1 and abc.txt will be concatenated and send to file2 $cat file1 > /dev/lp0 The contents of file file1 will be sent to printer instead of console
Output Redirection without overwriting	>>	In output redirection the file is cleared before writing to it. The >> is used so that output is appended and not overwritten	$cat file1 > file1 The file1 contents will be cleared $cat file2 >> file2 The file2 will have its contents appended to it
Pipe	|	The pipe character | is used between two commands so that output of first command is send as input to the second command	$ ls –l | grep "abc" Displays the line in the output of ls –l containing pattern abc

Filters in Linux

When a program takes its input from another program, performs some operation on that input, and writes the result to the standard output (which may be piped to yet another program), it is referred to as a filter. Linux has a number of filters.

- grep: Find lines in stdin that match a pattern and print them to stdout.
- sort: Sort the lines in stdin, and print the result to stdout.
- uniq: Read from stdin and print unique (that are different from the adjacent line) to stdout.
- cat: Read lines from stdin (and more files), and concatenate them to stdout.
- more: Read lines from stdin, and provide a paginated view to stdout.
- cut: Cut specified byte, character or field from each line of stdin and print to stdout.
- paste: Read lines from stdin (and more files), and paste them together line-by-line to stdout.
- head: Read the first few lines from stdin (and more files) and print them to stdout.
- tail: Read the last few lines from stdin (and more files) and print them to stdout.
- wc: Read from stdin, and print the number of newlines, words, and bytes to stdout.
- tr: Translate or delete characters read from stdin and print to stdout.

In previous assignment, cat ,head, tail, more , pg, tr etc are covered.
Some of the most commonly used filters are explained below**:**

- wc
- sort
- unique
- grep
- cut
- paste

wc command
wc stands for word count. As the name implies, it is mainly used for counting purpose.
It is used to find out number of lines, word count, byte and characters count in the files specified in the file arguments .By default it displays four-columnar output.First column shows number of lines present in a file specified, second column shows number of words present in the file, third column shows number of characters present in file and fourth column itself is the file name which are given as argument
Syntax:
wc [OPTION]... [FILE]...

Options :
1. -l: This option prints the number of lines present in a file. With this option wc command displays two-columnar output, 1st column shows number of lines present in a file and 2nd itself represent the file name.
2. -w: This option prints the number of words present in a file. With this option wc command displays two-columnar output, 1st column shows number of words present in a file and 2nd is the file name.
3. -c: This option displays count of bytes present in a file. With this option it display two-columnar output, 1st column shows number of bytes present in a file and 2nd is the file name.
4. -L: The ‘wc’ command allow an argument -L, it can be used to print out the length of longest (number of characters) line in a file.

```
$ cat state.txt
Andhra Pradesh
Arunachal Pradesh
Assam
Bihar
Chhattisgarh

$ wc state.txt
 5  7 63 state.txt

$ wc -l state.txt
5 state.txt

$ wc -w state.txt
7 state.txt

$ wc -c state.txt
63 state.txt
```

```
$ wc -L state.txt
17 state.txt
```

SORT command

SORT command is used to sort a file, arranging the records in a particular order. By default, the sort command sorts file assuming the contents are ASCII. Using options in sort command, it can also be used to sort numerically.

Syntax :

sort [options] [files]

Sort Options:

Some of the options supported are:

sort -b: Ignore blanks at the start of the line.

sort -r: Reverse the sorting order.

sort -o: Specify the output file.

sort -n: Use the numerical value to sort.

sort -M: Sort as per the calendar month specified.

sort -u: Suppress lines that repeat an earlier key.

sort -k POS1, POS2: Specify a key to do the sorting. POS1 and POS2 are optional parameters and are used to indicate the starting field and the ending field indices. Without POS2, only the field specified by POS1 is used. Each POS is specified as "F.C" where F represents the field index, and C represents the character index from the start of the field.

sort -t SEP: Use the provided separator to identify the fields.

Examples

Assume the below initial contents of file1.txt for the following examples

```
01 Priya
04 Shreya
03 Tuhina
02 Tushar
```

Sort with default ordering:

```
$ sort file1.txt
01 Priya
02 Tushar
03Tuhina
04 Shreya
```

In this example, the sorting is first performed using the first character. Since this is the same for all lines, the sorting then proceeds to the second character. Since the second character is unique for each line, the sorting ends there.

Sort in reverse ordering:

```
$ sort -r file1.txt
04 Shreya
03Tuhina
02 Tushar
01 Priya
```

In this example, the sorting is done similar to the above example, but the result is in the reverse order.

Sort by the second field:

```
$ sort -k 2 file1.txt
01 Priya
04Shreya
03Tuhina
02 Tushar
```

Now assume the original file2.txt is as below

```
01 Priya
01 Pooja
01 Priya
01 Pari
```

Sort with default ordering

```
$ sort file2.txt
01 Pari
01 Pooja
01Priya
01Priya
```

Sort suppressing repeated lines

```
$ sort -u file2.txt
01 Pari
01 Pooja
01Priya
```

<u>Uniq Command</u>

The uniq command in Linux is a command line utility that reports or filters out the repeated lines in a file. In simple words, uniq is the tool that helps to detect the adjacent duplicate lines and also deletes the duplicate lines. uniq filters out the adjacent matching lines from the input file(that is required as an argument) and writes the filtered data to the output file .

Syntax of uniq Command :

```
$uniq [OPTION] [INPUT][OUTPUT]]
```

The syntax of this is quite easy to understand. Here, INPUT refers to the input file in which repeated lines need to be filtered out and if INPUT isn't specified then uniq reads from the standard input. OUTPUT refers to the output file in which you can store the filtered output generated by uniq command and as in case of INPUT if OUTPUT isn't specified then uniq writes to the standard output.

The options of uniq command are:
c : Count of occurrence of each line.
d : Prints only duplicate lines.
D : Print all duplicate lines
f : Avoid comparing first N fields.
i : Ignore case when comparing.

s : Avoid comparing first N characters.
u : Prints only unique lines.
w : Compare no more than N characters in lines

Uniq Command Examples:
First create the following example.txt file in your unix or linux operating system.

```
> cat example.txt
Unix operating system
unix operating system
unix dedicated server
linux dedicated server
```

1. Suppress duplicate lines
The default behavior of the uniq command is to suppress the duplicate line. Note that, you have to pass sorted input to the uniq, as it compares only successive lines.

```
$ uniq example.txt
unix operating system
unix dedicated server
linux dedicated server
```

If the lines in the file are not in sorted order, then use the sort command and then pipe the output to the uniq command.

```
$sort example.txt | uniq
```

2. Count of lines.
The -c option is used to find how many times each line occurs in the file. It prefixes each line with the count.

```
$ uniq -c example.txt
    2 unix operating system
    1 unix dedicated server
    1 linux dedicated server
```

3. Display only duplicate lines.
You can print only the lines that occur more than once in a file using the -d option.

```
$ uniq -d example.txt
unix operating system
```

```
$ uniq -D example.txt
unix operating system
unix operating system
```

The -D option prints all the duplicate lines.

4. Skip first N fields in comparison.
The -f option is used to skip the first N columns in comparison. Here the fields are delimited by the space character.

```
$uniq -f2 example.txt
unix operating system
unix dedicated server
```

In the above example the uniq command, just compares the last fields. For the first two lines, the last field contains the string "system". Uniq prints the first line and skips the second. Similarly it prints the third line and skips the fourth line.

5. Print only unique lines.
You can skip the duplicate lines and print only unique lines using the -u option

```
$uniq -u example.txt
unix dedicated server
linux dedicated server
```

grep command
The grep filter searches a file for a particular pattern of characters, and displays all lines that contain that pattern. The pattern that is searched in the file is referred to as the regular expression (grep stands for globally search for regular expression and print out).
Syntax:
grep [options] pattern [files]
Options Description
-c : This prints only a count of the lines that match a pattern
-h : Display the matched lines, but do not display the filenames.
-i : Ignores, case for matching
-l : Displays list of a filenames only.
-n : Display the matched lines and their line numbers.
-v : This prints out all the lines that do not matches the pattern
-e exp : Specifies expression with this option. Can use multiple times.
-f file : Takes patterns from file, one per line.
-E : Treats pattern as an extended regular expression (ERE)
-w : Match whole word
-o : Print only the matched parts of a matching line,with each such part on a separate output line.
Sample Commands

Consider the below file as an input.

```
$cat > geekfile.txt
unix is great os. unix is opensource. unix is free os.
learn operating system.
Unix linux which one you choose.
uNix is easy to learn.unix is a multiuser os.Learn unix .unix is a powerful.
```

1. Case insensitive search : The -i option enables to search for a string case insensitively in the give file. It matches the words like “UNIX”, “Unix”, “unix”.

```
$grep -i "UNix" geekfile.txt
Output:
unix is great os. unix is opensource. unix is free os.
Unix linux which one you choose.
```

```
uNix is easy to learn.unix is a multiuser os.Learn unix .unix is a powerful.
```

2. Displaying the count of number of matches : We can find the number of lines that matches the given string/pattern

```
$grep -c "unix" geekfile.txt
Output:
2
```

3. Display the file names that matches the pattern : We can just display the files that contains the given string/pattern.

```
$grep -l "unix" *
or
 $grep -l "unix" f1.txt f2.txt f3.xt f4.txt
Output:
geekfile.txt
```

4. Checking for the whole words in a file : By default, grep matches the given string/pattern even if it found as a substring in a file. The -w option to grep makes it match only the whole words.

```
$ grep -w "unix" geekfile.txt
Output:
unix is great os. unix is opensource. unix is free os.
uNix is easy to learn.unix is a multiuser os.Learn unix .unix is a powerful.
```

5. Displaying only the matched pattern : By default, grep displays the entire line which has the matched string. We can make the grep to display only the matched string by using the -o option.

```
$ grep -o "unix" geekfile.txt
Output:
unix
unix
unix
unix
unix
unix
```

6. Show line number while displaying the output using grep -n : To show the line number of file with the line matched.

```
$ grep -n "unix" geekfile.txt
Output:
1:unix is great os. unix is opensource. unix is free os.
4:uNix is easy to learn.unix is a multiuser os.Learn unix .unix is a powerful.
```

7. Inverting the pattern match : You can display the lines that are not matched with the specified search sting pattern using the -v option.

```
$ grep -v "unix" geekfile.txt
Output:
learn operating system.
```

Unix linux which one you choose.
8. Matching the lines that start with a string : The ^ regular expression pattern specifies the start of a line. This can be used in grep to match the lines which start with the given string or pattern.

```
$ grep "^unix" geekfile.txt
Output:
unix is great os. unix is opensource. unix is free os.
```

9. Matching the lines that end with a string : The $ regular expression pattern specifies the end of a line. This can be used in grep to match the lines which end with the given string or pattern.

```
$ grep "os$" geekfile.txt
```

10.Specifies expression with -e option. Can use multiple times :

```
$grep –e "Agarwal" –e "Aggarwal" –e "Agrawal" geekfile.txt
```

11. -f file option Takes patterns from file, one per line.

```
$cat pattern.txt
Agarwal
Aggarwal
Agrawal

$grep –f pattern.txt  geekfile.txt
```

cut command

The cut command in UNIX is a command for cutting out the sections from each line of files and writing the result to standard output. It can be used to cut parts of a line by byte position, character and field. Basically the cut command slices a line and extracts the text. It is necessary to specify option with command otherwise it gives error. If more than one file name is provided then data from each file is not precedes by its file name.

Syntax:

```
cut OPTION... [FILE]...
```

When using the cut command you must use one and only one of the following options:

- -f (--fields=LIST) - Select by specifying a field, a set of fields, or a range of fields. This is the most commonly used option.
- -b (--bytes=LIST) - Select by specifying a byte, a set of bytes, or a range of bytes.
- -c (--characters=LIST) - Select by specifying a character, a set of characters, or a range of characters.
- Other options are:

- -d (--delimiter) - Specify a delimiter that will be used instead of the default “TAB” delimiter.
- --complement - complement the selection. When using this option cut will display all bytes, characters or fields except the selected

Let us consider two files having name state.txt and capital.txt contains 5 names of the Indian states and capitals respectively.

```
$ cat state.txt
Andhra Pradesh
Arunachal Pradesh
Assam
Bihar
Chhattisgarh
```

Without any option specified it displays error.

```
$ cut state.txt
cut: you must specify a list of bytes, characters, or fields
Try 'cut --help' for more information..
```

List with ranges

```
$ cut -b 1-3,5-7 state.txt
Andra
Aruach
Assm
Bihr
Chhtti
$ cut -c 2,5,7 state.txt
nr
rah
sm
ir
hti
```

Above cut command prints second, fifth and seventh character from each line of the file.

```
$ cut -f 1 state.txt
Andhra Pradesh
Arunachal Pradesh
Assam
Bihar
Chhattisgarh
```

If -d option is used then it considered space as a field separator or delimiter:

```
$ cut -d " " -f 1 state.txt
Andhra
Arunachal
Assam
Bihar
Chhattisgarh
```

Command prints field from first to fourth of each line from the file.

Command:

```
$ cut -d " " -f 1-4 state.txt
```

Output:

```
Andhra Pradesh
Arunachal Pradesh
Assam
Bihar
Chhattisgarh

$ cut --complement -d " " -f 1 state.txt
Pradesh
Pradesh
Assam
Bihar
Chhattisgarh
```

Paste Command

Paste command is one of the useful commands in Unix or Linux operating system. It is used to join files horizontally (parallel merging) by outputting lines consisting of lines from each file specified, separated by tab as delimiter, to the standard output. When no file is specified, or put dash ("-") instead of file name, paste reads from standard input and gives output as it is until a interrupt command [Ctrl-c] is given.

Syntax:

```
paste [OPTION]... [FILES]...
```

The options of paste command are:

- -d : Specify of a list of delimiters.
- -s : Paste one file at a time instead of in parallel.
- --version : version information
- --help : Help about the paste command.

Let us consider three files having name state, capital and number. state and capital file contains 5 names of the Indian states and capitals respectively. number file contains 5 numbers.

```
$ cat state
Arunachal Pradesh
Assam
Andhra Pradesh
Bihar
Chhattisgrah
$ cat capital
Itanagar
Dispur
Hyderabad
Patna
Raipur
```

Without any option paste merges the files in parallel. The paste command writes corresponding lines from the files with tab as a deliminator on the terminal.

```
$ paste number state capital
1       Arunachal Pradesh       Itanagar
```

```
2       Assam   Dispur
3       Andhra Pradesh  Hyderabad
4       Bihar   Patna
5       Chhattisgrah    Raipur
```
In the above command three files are merges by paste command.

```
$ paste -d "|" number state capital
1|Arunachal Pradesh|Itanagar
2|Assam|Dispur
3|Andhra Pradesh|Hyderabad
4|Bihar|Patna
5|Chhattisgrah|Raipur
```

More than one character is specified
```
$ paste -d "|," number state capital
1|Arunachal Pradesh,Itanagar
2|Assam,Dispur
3|Andhra Pradesh,Hyderabad
4|Bihar,Patna
5|Chhattisgrah,Raipur
```

First and second file is separated by '|' and second and third is separated by ','.
After that list is exhausted and reused.

2**. -s (serial):** We can merge the files in sequentially manner using the -s option. It reads all the lines from a single file and merges all these lines into a single line with each line separated by tab. And these single lines are separated by newline.

```
$ paste -s number state capital
1       2       3       4       5
Arunachal Pradesh       Assam   Andhra Pradesh  Bihar   Chhattisgrah
Itanagar        Dispur  Hyderabad       Patna   Raipur
```

In the above command, first it reads data from number file and merge them into single line with each line separated by tab. After that newline character is introduced and reading from next file i.e. state starts and process repeats again till all files are read.

Combination of -d and -s: The following example shows how to specify a delimiter for sequential merging of files:

```
$ paste -s -d ":" number state capital
1:2:3:4:5
Arunachal Pradesh:Assam:Andhra Pradesh:Bihar:Chhattisgrah
Itanagar:Dispur:Hyderabad:Patna:Raipur
```

Exercises:

Set A

1. Create any text file and count number of bytes, words, lines and length of longest line in file.
2. Count number of files in current working directory.
3. Redirect output of long listing of directories in abc.txt
4. Count number of users who are logged in and store output in a.txt
5. Displays a list of directories and how much space they consume, sorted from the largest to the smallest.
6. Concatenate two files a.txt , b.txt into third file c.txt

Set B

1. Create the following text file a.txt and write commands based on it.

```
Unix distributed 05 server
Linux virtual 3 server
Unix distributed 05 server
Distributed processing 6 system
```

a) Sort the above file in alphabetical order.
b) Sort the above file in descending order.
c) Sort the above file in numerical order.
d) Sort the above file on second field.
e) Sort the above file on second and fourth filed on reverse order.
f) Remove duplicate entries from above file and store in c.txt

2. Create the following text file a.txt and write commands based on it

```
welcome to ostechnix
welcome to ostechnix
Linus is the creator of Linux.
Linux is secure by default
Linus is the creator of Linux.
Top 500 super computers are powered by Linux
```

a) Write a linux command to remove consecutive duplicate lines.
b) Write a linux command to display only uniq lines.
c) Write a linux command to display only duplicate lines
d) Write a linux command to display number of occurrences of each line in a file.
e) Write a linux command to display limit the comparison to first 4 characters of lines in a file and display the repeated lines.
f) Write a linux command to avoid the comparison with the first 4 characters of lines in a file.

3. Create the following text file a.txt and write commands based on it

```
This is line 1 UNIX UNIX
This is line 2 unix
This is line 3 Unix Unix
This is line 4 hello
```

a) Write a linux command to display lines that search pattern “unix” .
b) Write a linux command to display lines that search pattern “unix” in case-insensitive manner .
c) Write a linux command to display line numbers that search pattern “unix”
d) Write a linux command to display count of lines that search pattern “unix”
e) Write a linux command to display lines that does contain pattern “unix”
f) Write a linux command to display lines that contains starting letter U and ending with letter x.

<u>Set C</u>

1. Create file as follows and write commands for same.

```
$ cat file.txt
unix or linux os
is unix good os
is linux good os
```

a) Write a linux command to print characters of 4th position.
b) Write a linux command to print characters by range 4-7.
c) Write a linux command that prints the second field in each line by treating the space as delimiter.

2. Create file as follows and write commands for same.

```
Linux
Unix
Solaris
HPUX
AIX
```

a)Write a linux command to join all lines separated by tab.
b)Write a linux command to join all lines separated by comma.
c)Write a linux command to merge a file by pasting the data into 2 columns
d)Write a linux command to merge a file by pasting the data into 2 columns using a colon separator

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well done [ ]

Signature of the instructor: _________________ Date: _____________________

Assignment 5

Aim: To study File permission, Process Management, Device Configuration, Network Configuration and System Security Linux Commands.

Pre-requisite: Knowledge of Basic Linux Commands.

The student should read following topics before starting exercise.

❖ File Permissions

Linux is a multi-user operating system. In Multi user environment more than one user can access one operating system. So practically there is requirement to protect the users from each other. Linux provides different types of commands to prevent access from unauthorized user. These commands are related to set permissions for file.

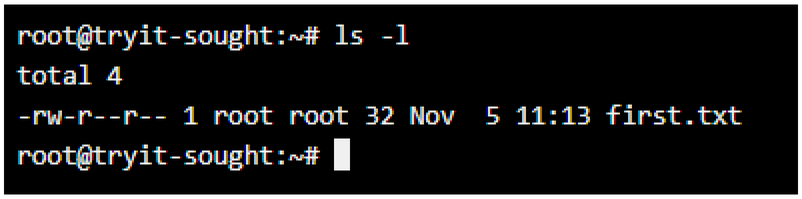
In Linux operating system, each file and directory has**3 types of owner**:

- **User:**A user is the owner of the file. By default, the person who created a file becomes its owner. Hence, a user is also sometimes called an owner.
- **Group**: A user- group can contain multiple users. All users belonging to a group will have the same access permissions to the file. Suppose you have a project where a number of people require access to a file. Instead of manually assigning permissions to each user, you could add all users to a group, and assign group permission to file such that only this group members and no one else can read or modify the files.
- **Other**:Any other user who has access to a file. This person has neither created the file, nor belongs to a user or group who could own the file. Practically, it means everybody else. Hence, when you set the permission for others, it is also referred as set permissions for the world.

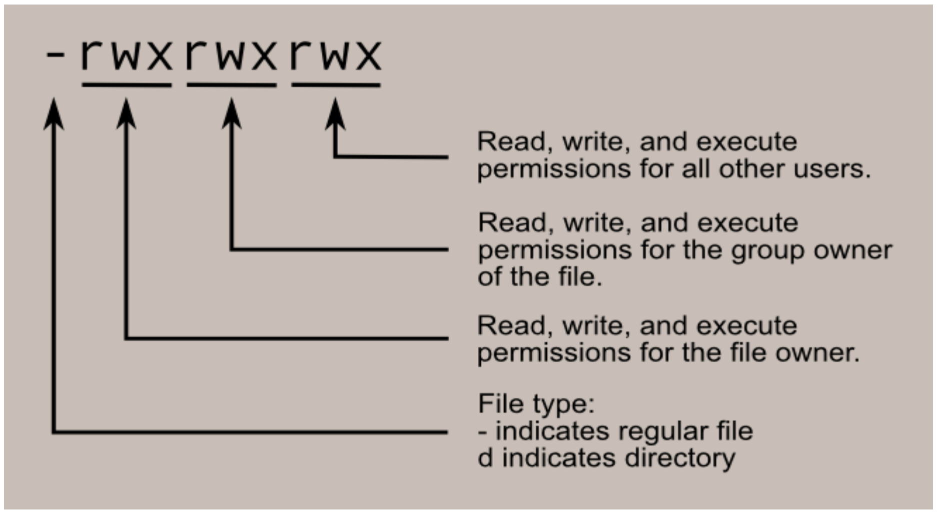
Every file and directory in your Linux system has following **3 types permissions** defined for all the 3 owners discussed above:

- **Read (r):** This permission give you the authority to open and read a file. Read permission on a directory gives you the ability to lists it's content.
- **Write (w):** The write permission gives you the authority to modify the contents of a file. The write permission on a directory gives you the authority to add, remove and rename files stored in the directory.
- **Execute (x):**In Linux, you cannot run a program unless the execute permission is set. If the execute permission is not set, you might still be able to see/modify the program code(provided read & write permissions are set), but cannot run program.

To check the permission for file by typing ls –l command


```
root@tryit-sought:~# ls -l
total 4
-rw-r--r-- 1 root root 32 Nov  5 11:13 first.txt
root@tryit-sought:~#
```

We will see how the first portion of ls command is interpreted. It consists of a character indicating the file type, followed by three sets of three characters that conveythe reading, writing and execution permission for the owner, group, and other.


- rwx rwx rwx
Read, write, and execute permissions for all other users.
Read, write, and execute permissions for the group owner of the file.
Read, write, and execute permissions for the file owner.
File type:
- indicates regular file
d indicates directory

Following are 3 commands to change File Permissions:

1. **chmod :**chmodstands for 'change mode'. Using the command, we can set permissions (read, write, execute) on a file/directory for the owner, group and other.

Syntax:

chmod permissions filename

There are 2 ways to use this command.

  a. **Absolute(Numeric) mode**

In this file permissions are not represented using 3 digit octal number.

Number in Octal	**umber in Binary**	**Permission Type**	**Meaning**
0	000	No Permission	---
1	001	Execute	--x
2	010	Write	-w-
3	011	Write + Execute	-wx
4	100	Read	r--
5	101	Read + Execute	r-x
6	110	Read + Write	rw-
7	111	Read + Write + Execute	rwx

Here are **examples** of chmod using Absolute mode.

- chmod 766 first.txt

  Before chmod first.txt file has – rw- r- -r- -permission,
  Read + Write for Owner, Read for Group and other
  After chmod command,
  Read + Write + Execute for Owner, Read + Write for Group and other
  We can check this by using ls -l command.
- chmod 711 first.txt

Read + Write + Execute for Owner, Execute for Group and other

- chmod –R 777 fybca

Here fybca is a directory. We are setting Read + Write + Execute. –R is used to set permissions recursively for subdirectories of fybca.

NOTE:
Becoming a Super user for a short while
It is often necessary to become the superuser to perform important system administration tasks, but as you have been warned, you should not stay logged in as the superuser. There is a program that can give you temporary access to the superuser's privileges. This program is called **su (short for substitute user)** and can be used in those cases when you need to be the superuser for a small number of tasks. To become the superuser, simply type the su command. You need to typesuperuser's password. After executing the su command, you have a new shell session as the superuser. To exit the superuser session, type exit and you will return to your previous session.

```
[me@linuxbox me]$ su
Password:
[root@linuxbox me]#
```

In the Absolute mode, you change permissions for all 3 owners. In the symbolic mode, you can modify permissions of a specific owner. It makes use of mathematical symbols to modify the file permissions.

Operator	Description
+	lds a permission to a file or directory
-	moves the permission
=	ts the permission and overrides the permissions set earlier.

The various owners are represented as -

u User/Owner
g Group
o Other
a all

Here are some **examples** using Symbolic mode

- chmodo+r first.txt

Beforechmod

```
[root@linuxbox me]$ls -l
   -rwx - - x- -x 1 root root 16 NOV 6 10:19 first.txt
```

After chmod

```
[root@linuxbox me]$ ls -l
   -rwx r—x r—x 1 root root 16 NOV 6 10:19 first.txt
```

- chmod a-x first.txt

Removing Execute permission for all users

2. **chown :**chown means change owner. You can change the owner of a file by using the chown command.In order to change the owner of a file, you must be the superuser. To do this use su command, then we executed chown, and finally we typed exit to return to our previous session.

Example:

```
[me@linuxbox me]$ su
   Password:
   [root@linuxbox me]# chown fybca12 first.txt
   [root@linuxbox me]# exit
   [me@linuxbox me]$
```

Beforechown

```
[root@linuxbox me]$ ls -l
   -rwx - - x- -x 1 rootroot 16 NOV 6 10:19 first.txt
```

Afterchown

```
[root@linuxbox me]$ ls -l
   -rwx - - x- -x 1 fybca12 root 16 NOV 6 10:19 first.txt
```

3. **chgrp :**The group ownership of a file or directory may be changed with chgrp.

Example:

```
[me@linuxbox me]$ su
  Password:
  [root@linuxbox me]# chgrp fybca12 first.txt
  [root@linuxbox me]# exit
  [me@linuxbox me]$
```

Before chgrp

```
  [root@linuxbox me]$ ls -l
  -rwx - - x- -x 1 fybca12root 16 NOV 6 10:19 first.txt
```

Afterchgrp

```
[root@linuxbox me]$ ls -l
  -rwx - - x- -x 1 fybca12 fybca12 16 NOV 6 10:19 first.txt
```

❖ **Linux process management commands**

What is a Process?

An instance of a program is called a Process. In simple terms, any command that you give to your Linux machine starts a new process.

Types of Processes:

1. **Foreground Processes:** They run on the screen and need input from the user.

E.g. Office Programs

2. **Background Processes:** They run in the background and usually do not need user input. E.g. Antivirus program.

Following are the Linux commands for Process Management:

Command	Usage	Example
bg	To run process in background. If you start a foreground program/process from the terminal, then you cannot work on the terminal, till the program is up and running. A particular task may take lots of processing power and may even take hours to complete. You do not want your terminal to be held up for such a long time. To avoid such a situation, you can run the program and send it to the background so that terminal remains available to you using bg command.	`[root@linuxbox me]# gedittry.c` `[root@linuxbox me]# bg` bg command moves current process in background.

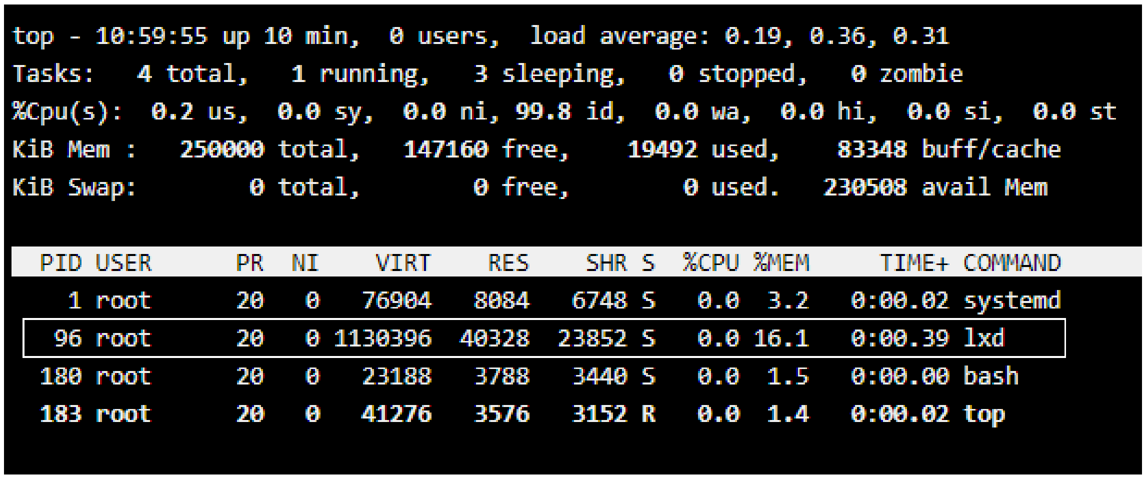
fg	It is used to continue a program which was stopped and bring as foreground process.	`[root@linuxbox me]# fg` command moves last process to foreground.

ps	This command stands for 'Process Status'. It is similar to the "Task Manager" that pop-ups in a Windows Machine when we use Ctrl+Alt+Del. This command is similar to 'top' command but the information displayed different way.	`root@tryit-hip:~# ps`
		`PID  TTY    TIME       CMD`
		`176    ?   0:00:00        bash`
		`184     ? 00:00:00  ps`
		`196     ?              00:00:05`
		`firefox`
		
		You can also check the process **status of a singleprocess**,
		
		`root@tryit-hip:~# ps 176`
		`PID      TTY      STAT`
		`TIMECOMMAND`
		`176      ?           Ss 0:00`
		`bash`

kill	The kill command can terminates a process for specified process ID	`root@tryit-hip:~# kill 196` Where 196 is Proccess ID of firefox
free	This command shows the free and used memory (RAM) on the Linux system.	`root@tryit-hip:~# free` free -m to display output in MB free -g to display output in GB

nice	The highest nice value is 19, which inversely gives process the lowest nice priority. The lowest nice value is -20, which inversely gives a process the highest nice priority.
	
	The default value of all the processes is 0.
	
	To change priority of process
	`nice  -n   'Nice value' process name`
		`root@tryit-hip:~# nice  -n  19 bash`
		
		`It will change the priority of bash process to 19 means lower priority.`

- **top:** This utility tells the user about all the running processes on the Linux machine.



```
top - 10:59:55 up 10 min,  0 users,  load average: 0.19, 0.36, 0.31
Tasks:   4 total,   1 running,   3 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.2 us,  0.0 sy,  0.0 ni, 99.8 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :   250000 total,   147160 free,    19492 used,    83348 buff/cache
KiB Swap:        0 total,        0 free,        0 used.   230508 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
    1 root      20   0   76904   8084   6748 S   0.0  3.2   0:00.02 systemd
   96 root      20   0 1130396  40328  23852 S   0.0 16.1   0:00.39 lxd
  180 root      20   0   23188   3788   3440 S   0.0  1.5   0:00.00 bash
  183 root      20   0   41276   3576   3152 R   0.0  1.4   0:00.02 top
```

Press 'q' on the keyboard to move out of the process display.

Field	**Description**	**Example**
PID	The process ID of each task	96
User	The username of task owner	root
PR	Priority Can be 20(highest) 20 or -20(lowest)	20
NI	The nice value of a task	0
VIRT	Virtual memory used (kb)	1130396
RES	Physical memory used (kb)	40328
SHR	Shared memory used (kb)	23852
S	Status There are five types: 'D' = uninterruptible sleep 'R' = running 'S' = sleeping 'T' = traced or stopped 'Z' = zombie	S
%CPU	% of CPU time	0.0
%MEM	Physical memory used	16.1
TIME+	Total CPU time	0.00.39
Command	Command name	lxd

- **at :**The **at** command schedules a command to be run once at a particular time**.** For scheduling a job using **at** command use shell script file. Specify commands in script file and run at particular time.

Example:

vi Hello.sh

echo "Hello"

Save Hello.sh

Now in terminal type :

```
[root@linuxbox me]# at Hello.sh now
```

It will show output immediately

```
[root@linuxbox me]# at Hello.sh 11:20
```

Output will be displayed at 11:20am

You can check schedule status of Hello.sh file using atq Command

```
[root@linuxbox me]# atq
12    2019-05-10 11:20 root
```

Linux Network Configurations commands

Command	Usage	Example
ifconfig	It is used to get IP address of the system	`[root@localhost]#ifconfig`

ping	PING (Packet INternet Groper) command is to test connectivity between two nodes. Ping use ICMP (Internet Control Message Protocol) to communicate to other devices. You can ping host name of ip address.	`[root@localhost]#ping 192.168.26.100`
		
		`[root@localhost]#ping www.google.com`

ftp	File Transfer Protocol is used for logging in and establishing a connection with a remote host. Uploading or Downloading file to/from remote host.
	Syntax:
	ipaddress or hostname
		`[root@localhost]#ftp 192.168.26.100`
		
		`Specify username and password`
		
		Use following commands to get data from remote server.
		
		`dir – Display listing of files in current directory`

		cd "dirname" - To change the directory put filename - Upload file from local to remote computer get filename –Download file from remote to local computer quit - logout Use passive command if you get any error of firewall
telnet	It helps to connect to a remote Linux computer, run programs remotely and conduct administration This utility is similar to the Remote Desktop. Syntax: telnet ipaddress or hostname	`[root@localhost]#telnet 192.168.26.100` Specify username and password Use logout for exit Now executelinux commands to access data of remote server. Commands are executed on remote machine and not on local computer.
ssh	SSH which stands for Secure Shell, It is used to connect to a remote computer securely. Compare to Telnet, SSH is secure wherein the client /server connection is authenticated using a digital certificate and passwords are encrypted. Hence it's widely used by system administrators to control remote Linux servers. Syntax: sshipaddress or hostname	`[root@localhost]#ssh 192.168.26.100` Specify username and password Use logout for exit Now execute linux commands to access data of remote server.

host	host performs DNS lookups, converting domain names to IP addresses and vice versa.	`[root@localhost]#hostwww.google.com`
		
		Displays the IPv4 and IPv6 address.
		`[root@localhost]#host 74.125.25.147`
		
		splays the domain name.

uuenode	It is used to encode contents of file. So that contents are secured.	`oot@localhost]#uuencodefirst.txt encode.txt`
		
		`rst.txt is a plain text file encoded to encode.txt`

uudecode	It is used to decode contents of file.	`root@localhost]#uudecodeencode.txt try.txt`
		
		`encode.txt is decoded and stored in try.txt`

❖ Linux commands to manage local accounts

These commands are used to manage local accounts or users like creating new user, modifying or deleting user account.

Command	Usage	Example

useradd	Create a new user or account	`[root@localhost]#useradd fybca12`
		
		`Add fybca12 user with default setttings`
		
		`[root@localhost]#useradd fybca12 –d /home/fybca12`
		
		`Create a new user's home dir in /home`
		
		`[root@localhost]#useraddfybca12 -s /bin/csh`
		
		`Set C Shell as the default login shell for the fybca12`

usermod	Modify user account	`[root@localhost]#usermod -d /home2/fybca12fybca12`
		
		`Create the new home Dir for fybca12 in /home2 & Move old Dir contents to this Dir.`
		
		`root@localhost]#usermod -p $1$d8 fybca12`
		
		`Set the new passwd for the fybca12`

userdel	delete a user account and user's related files	`[root@localhost]#userdel -r fybca12` Delete the user account together with user's home directory and all files inside it.

passwd	It is used to set/reset password for user account.
	
	You can set password as your own without login as root.
		`[fybca12@localhost]#passwd`
		
		Specify Password and Retype new password.
		
		`[fybca12@localhost]#passwd –d`
		
		Removes the password for account.

Exercise

SET A

Perform the commands on the system and write the commands.

1. Create three text files one.txt, two.txt, three.txt. Check the permissions of created files.
2. Change permission of one.txt as give read, write, execute access for owner, Read, write for group and only read for other.
3. Change permission of two.txt as read, write, and execute access for owner, group and other.
4. Change permission of three.txt as read, write access for owner, Read, Execute for group and only Execute for other.
5. Create directory 'Assignment'. Create two subdirectories 'Ass1', 'Ass2' under it. Check the permission of directory.
6. Change permission of 'Assignment' directory. Give read, read, write, and execute access for owner, group and other including subdirectories also.
7. Change the owner of one.txt to ________________(username)
8. Change the group of two.txt to ________________(username)

SET B

1. Check the processes that are running on the system. Write the details of at least 3 process below.
2. See the details of particular process by specifying Process ID. Write command and details of particular process.
3. Terminate ________________________ process.
4. Check the free and used memory status of system in MB and GB.
5. Change the priority of ______________ process to -5.
6. Execute top command and see the output.

7. Create trial.txt file. Type 3 echo statements, date, ls command in that file. Schedule the execution status of that file at 15:30 (ie 3.30pm)
8. Check the schedule status of trial.txt

SET C

1. Check the IP address of your system. Mention the command with IP address.
2. Check the system with IP address ____________________________ is running in LAN.
3. Write a command to domain name of IP address ___________________________ connected in LAN. Write domain name also.
4. Use ftp to upload one.txt to remote server.
5. Download ___________________________ file from remote server using ftp.

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well done [ ]

Signature of the instructor: ______________________ Date: ____________________

Assignment 6

Aim: By learning to use Vi editor can benefit user in creating scripts and editing files.
Prerequisites: Basic knowledge of Linux operating System.

What is Vi :
The Default editor that comes with the UNIX operating system is called Vi (Visual) Editor
Using the Vi Text Editor:

- Vi is a full-screen text editor that is almost universally available on UNIX-based computer systems.
- The vi editor is the most popular and commonly used Linux text editor
- There are also versions of Vi for the IBM PC (and compatibles) and the Macintosh.
- Vi is useful for editing program files, entering data, composing mail messages, and plain text editing.

To Get Into and Out Of Vi

- **To Start vi**

To edit a file with vi, type any of the following commands at the UNIX system prompt.

Sr.No	Command	Description
1	Vi	It creates a new, unnamed file
2	vi filename	Creates a new file if it already does not exist, otherwise opens an existing file.
3	vi -r filename	It recoversfile from system crash (this may not recover all of the changes you made to your file in your last editing session)
4	vi -R filename	It opens the existing file in Read Only mode

- *To Exit fromVi*

You can save and quit Vi editor from command mode. Before writing save or quit command you have to press colon **`(:).`** Colon allows you to give instructions to Vi.

Following commands are used to exit from Vi editor.

Sr.No	Command	Description
1	:q	Quit without saving
2	:wq	Save the file(write) and quit (i.e. save and quit)

3	:w	Save the file but keep it open
4	:w fname	Write to file called fname (save as)
5	Shift+zz	Save the file and quit
6	:q!	Quit without saving changes i.e. discard changes
7	:w!	Save (and write to non-writable file)

To exit from Vi, first ensure that you are in command mode. Now, type**:**wq and press enter. It will save and quit Vi.

Type **:wq** to save and exit the file.

<u>Important Points to Note</u>

The following points will add to your success with Vi −

- You must be in command mode to use the commands. (Press Esc twice at any time to ensure that you are in command mode.)
- You must be careful with the commands. These are case-sensitive.
- You must be in insert mode to enter text.

<u>Vimodes:</u>

Vi has two modes:

i) Commandmode: While in command mode, everything you type is executed as a command to edit your document. This mode enables you to perform administrative tasks such as saving the files, executing the commands, moving the cursor, cutting (yanking) and pasting the lines or words, as well as finding and replacing. In this mode, whatever you type is interpreted as a command.

ii) Input 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. While in input mode, everything you type is inserted into your document (including command mode commands).

NOTE: Vi always starts in the **command mode**. To enter text, you must be in the insert mode for which simply type **i**. To come out of the insert mode, press the **Esc** key, which will take you back to the command mode. Change from input mode to command mode, press the **Esc key.**

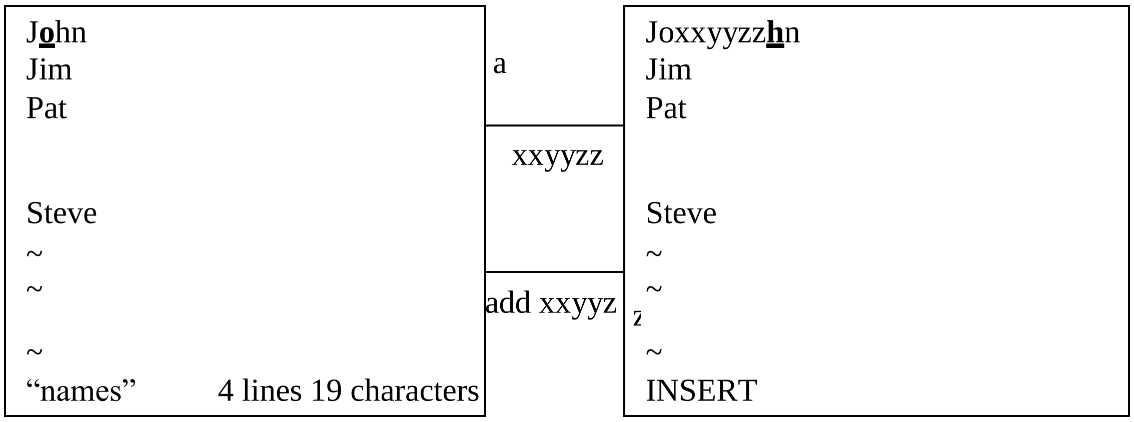
<u>Basic Vi Commands:</u>
There are several ways to change from command mode to input mode that is listed below.

1) **Entering input mode: (Editing Files)**

To edit the file, you need to be in the insert mode. There are many ways to enter the insert mode from the command mode –

Sr.No	**Command**	**Description**
1	i	enter input mode(text) before the current cursor location
2	I	enter input mode at the beginning of the current line
3	a	enter input mode after the current cursor location
4	A	enter input mode at the end of the current line
5	o	create a new line below the cursor location and enter input mode on it
6	O	create a new line above the cursor location and enter input mode on it

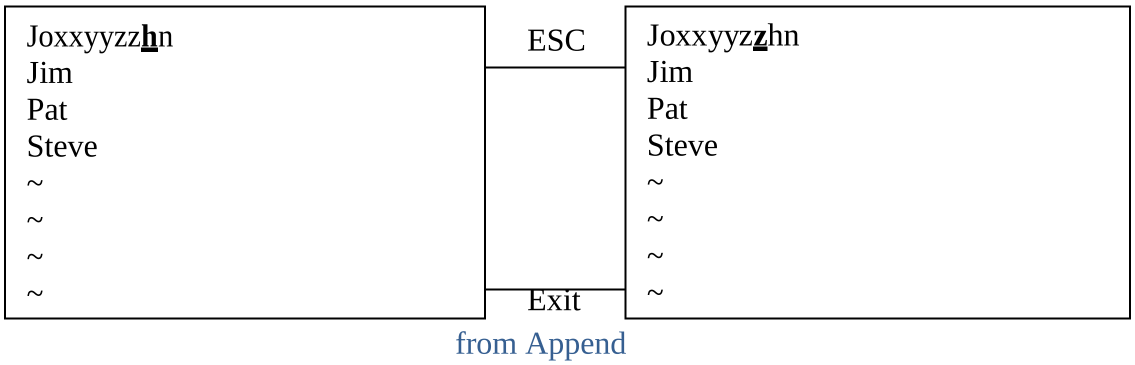
To add text you position the cursor over a character and press an “a”. This puts you in a special mode of operations called “insert mode”. Now everything typed is appended to the text *after* the character the cursor was positioned over:



```
John
Jim
Pat

Steve
~
~

~
“names”        4 lines 19 characters
```

```
Joxxyyzzhn
Jim
Pat

Steve
~
~

~
INSERT
```

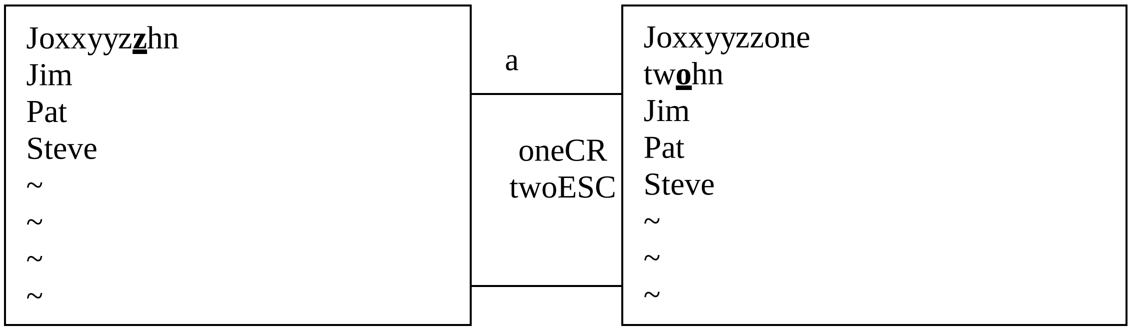
When you are done adding text, you press the ESC key. When you press ESC key, the cursor moves back to the last character you entered.



```
Joxxyyzzhn
Jim
Pat
Steve
~
~
~
~
```

```
Joxxyyzzhn
Jim
Pat
Steve
~
~
~
~
```

You can even put ↵RETURNs (CR) in the added text, and new lines appear.



```
Joxxyyzzhn
Jim
Pat
Steve
~
~
~
~
```

```
Joxxyyzzone
twohn
Jim
Pat
Steve
~
~
~
```

Embedded*CR*

The appending started between the z and hn of the first line, causing the hn to be carried to the next line when the CR i.e. (↵) was pressed.

1. Perform the following changes to your file. Specify the command and the resulting text as an answer.

Action	Command typed	Result
Change Jim in line 3 to Jirem		
Insert a new line “Tom and Jerry” after line number 3.		
Insert a new line at the end		

2) Deleting Characters and Words

Here is a list of important commands, which can be used to delete characters, wordsand lines in an open file −

Sr.No	**Command**	**Description**
1	x	Deletes the single character under the cursor location.
2	X	Deletes the character before the cursor location
3	Nx	Delete N Character,starting with character under cursor
4	dw	Deletes from the current cursor location to the next word
5	dNw	Deletes N words beginning with character under cursor
6	d^	Deletes from the current cursor position to the beginning of the line
7	d$	Deletes from the current cursor position to the end of the line
8	D	Deletes from the cursor position to the end of the current line
9	dd	Deletes the current lin.
10	3dd	It deletes 3 lines
11	dw	Delete word
12	4dw	Delete 4 words

1. Create a file in Vi editor. Perform the following operation by specifying the command and the resulting text asanswer.

Action (from current cursor position)	Command typed	Result
Delete 2 characters		
Delete 3 characters from 3rd line		
Delete 1stline		
Delete 4thline		

3) Copy, Paste, Undo and Repeat Commands

You can copy lines or words from one place and then you can paste them at another place using the following commands –

Sr.No	Command	Description
1	yy	Copies the current line.
2	Nyy	Copy the next N lines, including the current line, into the buffer.
2	yw	Copy one word
3	p(lowercase)	Paste the copied text after the cursor.
4	P(Uppercase)	Paste before the current line
5	9yy	Yank current line and 9 lines below.
6	U	Undo the last command
7	.(dot)	Repeat the last command

1. Create a file in Vi editor. Perform the following operation by specifying the command and the resulting text as answer.

Action (from current cursor position)	**Command typed**	**Result**
Cutline 2, 3 and put those after two line		
Copy line 3 and put it after line 4.		
Undo all the changes		
Locate the content at line 2		

4) Changing the text

Sr.No	Command	Description
1	r	Replace a single character under cursor
2	R	Overwrites multiple characters beginning with the character currently under the cursor. You must use **Esc** to stop the overwriting.
3	s	Replaces the current character with the character you type. Afterward, you are left in the insert mode.
4	S	Deletes the line the cursor is on and replaces it with the new text. After the new text is entered, vi remains in the insert mode.

5) Moving Cursor within a File:

To move around within a file without affecting your text, you must be in the command mode (press Esc twice). The following table lists out a few commands you can use to move around one character at a time –

Sr.No	Command	Description
1	k	Moves the cursor up one line
2	j	Moves the cursor down one line
3	h	Moves the cursor to the left one character position.
4	i	Moves the cursor to the right one character position

5	0(Zero)	Move Cursor to start of the current line(the one with the arrow)
6	$	Move Cursor to the end of the current line
7	w	Move cursor to beginning of next word
8	b	
9	:0<return> or 1G	Move cursor to first line in the file
10	:n<return>or nG	Move cursor to line ‘n’ in file
11	:$<return> or G	Move cursor to last line in file
12	nj, nk, nh, nl	move more than one column or line at a time

1. Create a file in Vi editor. Perform the following operation by specifying the command and the resulting text asanswer.

Action (from current cursor position)	Command typed	Result
Move _3_ lines down		
Move __4 columns right		
Move __4 columns left		
Move 3__ lines up		

6) Searching a String:

In command mode, with the help of **'/',** string can be searched in forward direction and with the help of **?** , string can be searched in backward direction. For example, **/abc** will do a forward search for string abc whereas? **abc** will do a backward search for string abc.

Sr.No	**Command**	**Description**
1	/string	String can be searched in forward direction
2	?string	String can be searched in backward direction
3	/^string	String can be searched in forward direction at the beginning of a line
4	/string$	String can be searched in forward direction at the end of a line
5	N	Go to the next occurrence of searched string
6	/\<he\>	Search for the word he (and not for there, here, etc.)
7	/pl[abc]ce	Search for place, plbce, and plcce

1. Create a fileMy_college in Vi editor. Perform the following operation by specifying the command and the resulting text asanswer.

Action (from current cursor position)	Command typed	Result
Search for keyword “College”		
Search keyword “College” in forward direction		

7) ReplacingText:

The substitution command (**:s/**) enables you to quickly replace words or groups of words within your files. Following is the syntax to replace text –

Syntax is - :s/old_string/new-string/g

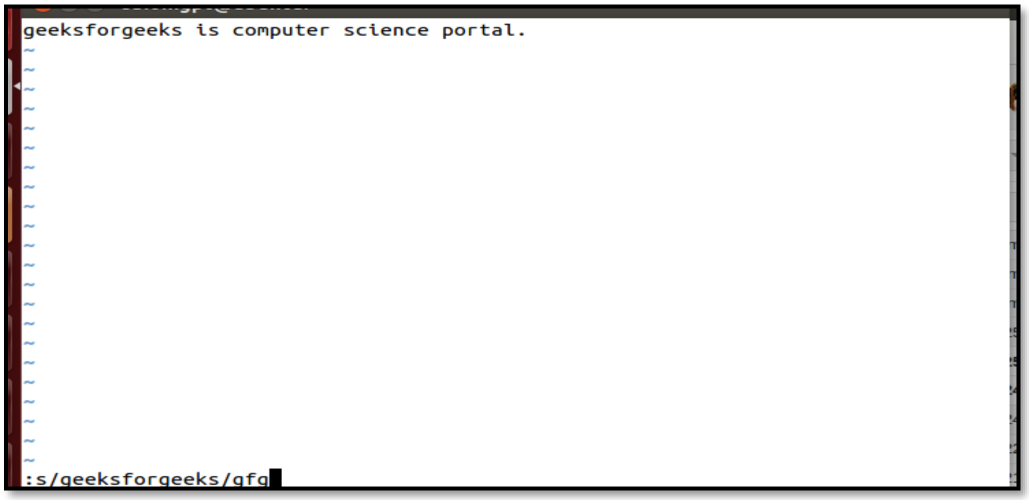
The **g** stands for globally. The result of this command is that all occurrences on the cursor's line are changed.

For example:

1) :s/java/os
Here “java” represents old string and “os” represents new string. It replaces each “java” string in a line with “os”string.

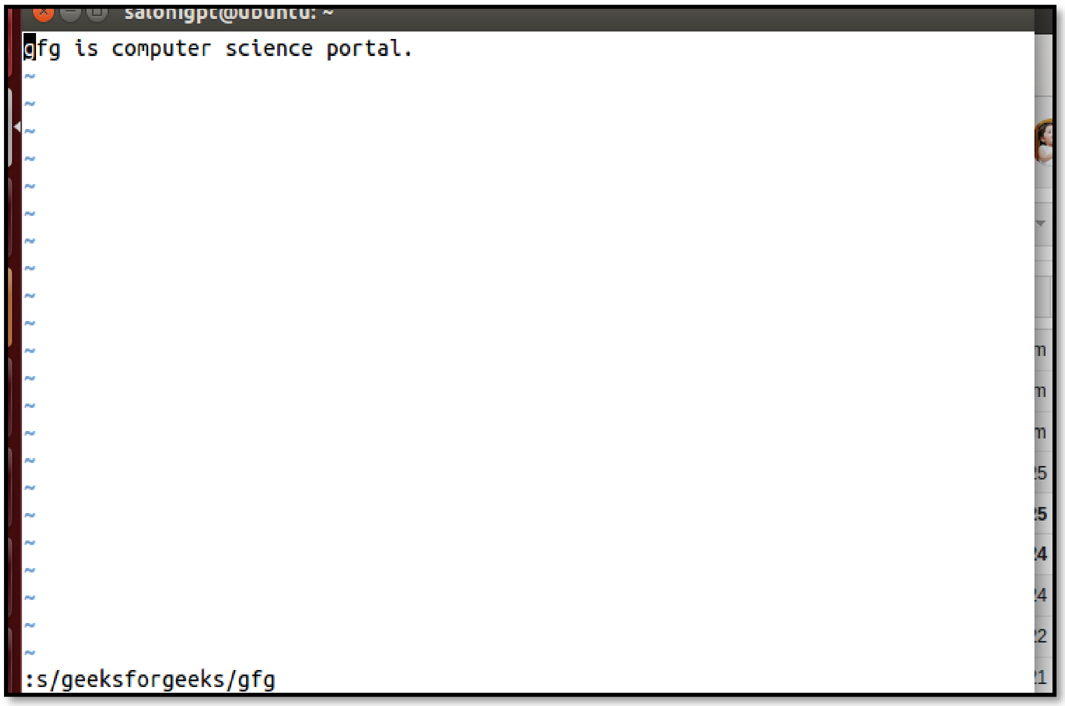
2) :s/geeksforgeeks/gfg/

Input Screen:



```
geeksforgeeks is computer science portal.
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
:s/geeksforgeeks/gfg
```

Output Screen



```
gfg is computer science portal.
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
:s/geeksforgeeks/gfg
```

3) :%s/pattern/replace/
   It replaces every occurrence of a string in the entire text.

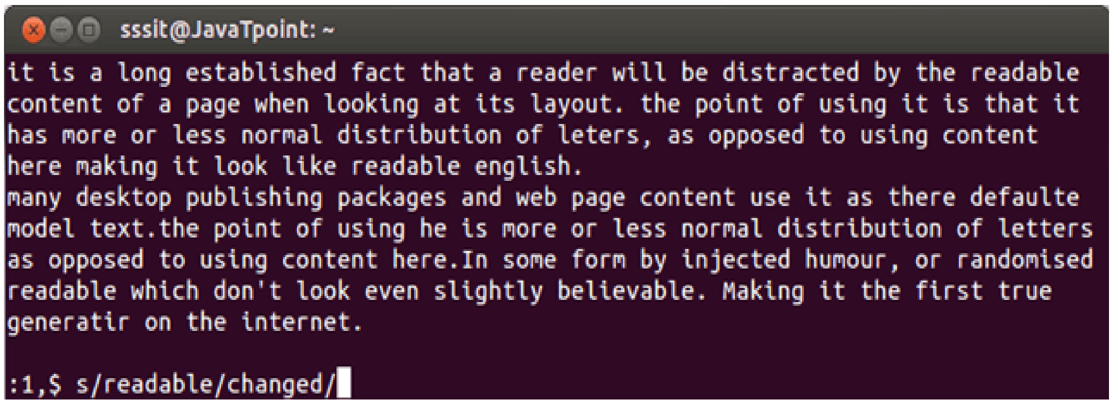
   For example -- **:%s/gfg/geeksforgeeks/**

4) :1,$ s/old_string/new string/
   It replaces the old string start from first line by new string to the last line.

   For Example -

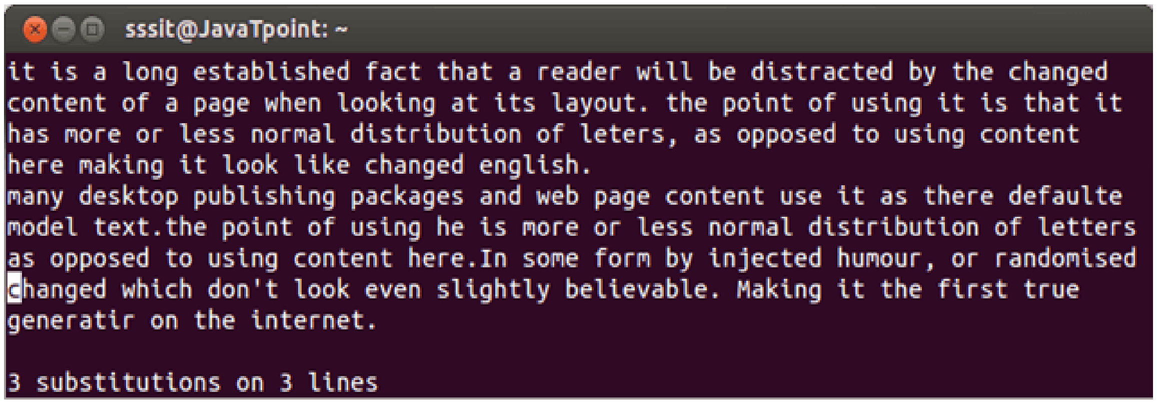
   : 1, $ s/readable/changed/

   Input Screen:



```
it is a long established fact that a reader will be distracted by the readable
content of a page when looking at its layout. the point of using it is that it
has more or less normal distribution of leters, as opposed to using content
here making it look like readable english.
many desktop publishing packages and web page content use it as there defaulte
model text.the point of using he is more or less normal distribution of letters
as opposed to using content here.In some form by injected humour, or randomised
readable which don't look even slightly believable. Making it the first true
generatir on the internet.

:1,$ s/readable/changed/
```

Output Screen:



```
it is a long established fact that a reader will be distracted by the changed
content of a page when looking at its layout. the point of using it is that it
has more or less normal distribution of leters, as opposed to using content
here making it look like changed english.
many desktop publishing packages and web page content use it as there defaulte
model text.the point of using he is more or less normal distribution of letters
as opposed to using content here.In some form by injected humour, or randomised
changed which don't look even slightly believable. Making it the first true
generatir on the internet.

3 substitutions on 3 lines
```

8) **Control Commands (Scrolling Commands) :** There are following useful commands which can used along with **Control Key**:

Sr.No	**Command**	**Description**
1	**CTRL+d**	Move Forward ½ screen or Scroll Down (half a screen)
2	**CTRL+f**	Move Forward one full screen
3	**CTRL+u**	Move Backward ½ screen or Scroll Up(half a screen)
4	**CTRL+b**	Move backward one full screen.
5	**CTRL+e**	Moves screen up one line.
6	**CTRL+y**	Moves screen down one line.

Exercise:

Set A:

1. Create a file by name Mycollege.txt with at least 25 lines long using vi editor's input commands -"a" and "i".Also try there place mode by examining the toggle feature of "i" character.

2. Create a file by name ______ with least25lineslongusingvieditor'sinputcommands–"a" and "i".Also try search command on the file.

Set B

1CreateafilebyInput devices.txt with least25lineslongusingvieditor'sinputcommands and try the following using Vi commands

a) Move to the first line of the file.
b) Insert a blank line below the 3rd line
c) Delete any 3 lines
d) Join any two lines together
e) Restore the single deleted line
f) Search the keyword "dev" from a file.

2. Create a file name_____containing five lines and execute the following set of commands

of vi editor and describe the result on thepaper.

Sr. No	Command
1	cc
2	D
3	C
4	s
5	S
6	rchr
7	R

Set C:

1Createafilebyname My_country.txt with least25lineslongusingVieditor'sinputcommands and try the following using Vi commands

a) Move to the first line of the file
b) Replace each occurrence of word "College" with Institute.
c) Move to the beginning of 2nd line.
d) Apply control command on file.
e) Search a keyword "India" overall in a file.
f) Copy and paste any 4 lines from a file.

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well done [ ]

Signature of the instructor: ______________________ Date: ________________________

Assignment 7

Aim: To study basic commands of shell programming.
Pre-requisite: Knowledge of Linux commands and vi/emac editor.

If you are using any major operating system you are indirectly interacting to shell. If you are running Ubuntu, Linux Mint or any other Linux distribution, you are interacting to shell every time you use terminal. So let us discuss about linux shells and shell scripting.
A shell in a Linux operating system takes input from you in the form of commands, processes it, and then gives an output. It is the interface through which a user works on the programs, commands, and scripts. A shell is accessed by a terminal which runs it.
There are mainly 4 important types of shells that are widely used.
They include:

- Bourne Shell (sh)
- C Shell (csh)
- Korn Shell (ksh)
- Bourne Again Shell (bash)

When you run the terminal, the Shell issues a command prompt (usually $), where you can type your input, which is then executed when you hit the Enter key. The output or the resultis thereafter displayed on the terminal.
Different statements used in shell script are,

Statement	**Usage**	**Example**
Read	Reading values from standard input	read first_name read salary
echo	To display output to user.	echo " What's your name?" a=10 echo "val = $a"

Expr	Write the result of the expression on the standard output. This command is primarily intended for arithmetic and string manipulation (convert string to integer.)	mult=\`expr $x * $y\`
		x=\`expr $x + 1\`

Test	Evaluates expression on its right or evaluates expression within square brackets	a=1; b=2; test $a –eq $b ; echo "*" read ch if [$ch =”y” –o $ch =”Y”] then …… Exit

Set	Assign a value to a shell variable (or multiple values to multiple variables). set without arguments displays the names and values of all shell variables, sorted by name	set val = 7 set new_val = “like seven” echo $new_val
Unset	Delete a shell variable. This does not just clear the variable, but makes it “not exist”. All variables are removed by "unset *	$a=”unset example” echo $a unset $a echo $a

Exercises

Set A:

1. Write a shell script to evaluate basic arithmetic operations.
2. Write a shell script to calculate simple interest.
3. Write a shell script to find area and perimeter of rectangle.
4. Write a shell script to calculate the gross salary.
5. Write a shell script to find area of circle by accepting input radius.

Set B:

1. Write a shell script to view contents of a file preceding with line numbers.
2. Write a shell script to show the list of users logged into the system.
3. Write a shell script to append the contents of one file to the end of another file.
4. Write a shell script to display file in reverse order.
5. Write a shell script to accept a file name and display number of words in a file. (use wc command)
6. Write a shell script to accept a directory name and display its contents. (use ls command)
7. Write a shell script to find size of given file.
8. Write a shell scriptto print out the length of longest (number of characters) line in a file.
9. Write a shell scriptto **display the file names that matches the given pattern.**
10. Write a shell script to accept a file name and a pattern. Display lines from the file in which the pattern is present along with line number. (use grep command)
11. Write a shell script to accept a name, and create a copy of it named as this name-(hypen)copy in the same directory . (use cp command)
12. Write a shell script that displays a list of files in current directory to which the user has read, write and execute permissions.

SET C:

1. Write a shell script to display the lines that are not matched with the specified search sting pattern.
2. Write a shell script to copy all files of the source directory to the destination directory.

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well done [ ]

Signature of the instructor: ______________________ Date: ________________________

Assignment 8

Aim: To study different conditional statements and command line arguments in shell programming.

Pre-requisite: Knowledge of Linux commands and vi/ emac editor.

The student should read following topics before starting exercises.

Conditional Statements:

Shell scripts often need to be constructed to execute different instructions depending on the value of specific control variables. The different paths of execution are specified using conditional statements. There are total 5 conditional statements which can be used in bash programming

- if statement
- if-else statement
- if..elif..else..fi statement (Else If ladder)
- if..then..else..if..then..fi..fi..(Nested if)
- switch statement

Conditional statement	**Syntax**	**Example**

if statement
if statement, if a particular test is true, then perform a given set of actions.
	if [expression]
	then
	  statement
	fi
		Script to check two variables are equal or not
		#Initializing two variables
		a=10
		b=20
		
		#Check whether they are equal
		if [$a == $b]
		then
		   echo "a is equal to b"
		fi
		
		#Check whether they are not equal
		if [$a != $b]
		then
		   echo "a is not equal to b"
		fi

if-else
If specified condition is not true in if part then else part will be execute.
	if [expression]
	then
	  statement1
	else
	  statement2
	fi
		Script to check if an input number is positive or not:
		echo "Enter a number"
		read num
		if [$num -gt 0]
		then
		   echo "It is a positive number" else
		   echo "It is not a positive integer"
		fi

if..elif..else..fi statement (Else If ladder)
use multiple conditions in one if-else block, then elif keyword is used in shell. If expression1 is true then it executes statement 1 and 2, and this process continues. If none of the condition is true then it processes else part
	if [expression1]
	then
	  statement1
	  statement2
	  .
	 elif [expression2]
	then
	  statement3
	  statement4
	  .
	else
	  statement
	 fi
		Script to check if an input number is positive, zero or negative:
		echo “Enter a number”
		read num
		if [$num -gt 0]
		then
		echo “It is a positive number” elif [$num -eq 0]
		then
		echo “num is equal to zero” else
		echo “It is not a positive integer”

if..then..else..if..then..fi..fi..(Nested if)
Nested if-else block can be used when, one condition is satisfies then it again checks another condition. In the syntax, if expression1 is false then it processes else part, and again expression2 will be check.
	if [expression1]
	 then
	 statement1
	 statement2
	  .
	 else
	 if [expression2]
	 then
	   statement3
	   .
	 fi
		Script to illustrate use of nested if else
		
		echo "Enter Your Country:"
		read cn
		
		if [$cn -eq 'India']
		then
		  echo "Enter Your State:"
		   read st
		  if [$st -gt 'Gujarat']
		   then
		      echo "Welcome to Gujarat"
		   elif
		      echo "You are Not Gujarati"
		   fi
		elif
		  echo "Other Country"
		fi

switch statement
case statement works as a switch statement if specified value match with the pattern then it will execute a block of that particular pattern.
When a match is found all of the associated statements until the double semicolon (;;) is executed.
A case will be terminated when the last command is
	case $[variable_name] in
	value1)
	   Statement 1
	   ;;
	value2)
	   Statement 2
	   ;;
	value3)
	   Statement 3
	   ;;
	value4)
	   Statement 4
	   ;;
	valueN)
		Script to illustrate Case Statement Example
		
		echo "Enter Country Code:"
		 read co
		
		 case $co in
		 'IN') echo "India"
		 ;;
		 'PK') echo "Pakistan"
		 ;;
		 *) echo "Enter Vailid Country Code"
		 ;;
		esac

executed. If there is no match, the exit status of the case is zero	Statement N ;; *) Default Statement ;; esac	

File-based condition
File-based conditions are unary expressions and often used to examine a status of a file.
	• -a file Returns true if file exists
	• -b file Returns true if file exists and is a block special file
	• -c file Returns true if file exists and is a character special file
	• -d file Returns true if file exists and is a directory
	• -e file Returns true if file exists
	• -r file Returns true if file exists and is readable
	• -s file Returns true if file exists and has a greater size that zero
	• -s file Returns true if file exists and has a greater size that zero
	• -w file Returns true if file exists and is writable
	• -x file Returns true if file exists and is executable
	• -N file Returns true if the file exists and has been modified since it was last read
		Script to illustrate file based conditions.
		`ls -l`
		`read -p "Enter a file name: " filename`
		`if [ -e $filename ]`
		`then`
		`  echo "file exists!"`
		`  if [ -r $filename ]`
		`  then`
		`      status="readable "`
		`  fi`
		`  if [ -w $filename ]`
		`  then`
		`      status=$status"writable "`
		`  fi`
		`  if [ -x $filename ]`
		`  then`
		`      status=$status"executable"`
		`  fi`
		`   echo "file permission: "$status`
		`else`
		`  echo "file does not exist"`
		`fi`

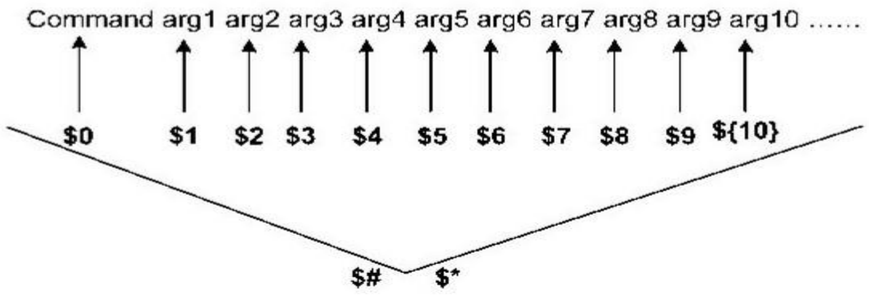
Arithmetic-based Condition	• -eq Equal
	• -ge Greater Than or Equal
	• -gt Greater Than
	• -le Less Than or Equal
	• -lt Less Than
	• -ne Not Equal
		Script to illustrate arithmetic based conditions
		`read -p "Enter an integer: " int1`
		`if [ $int1 -eq 0 ]`
		`then`
		`  echo "Zero"`
		`elif [ $int1 -lt 0 ]`
		`then`
		`  echo "Negative"`
		`else`
		`  if [ $((int1%2)) -eq 0 ]`
		`  then`

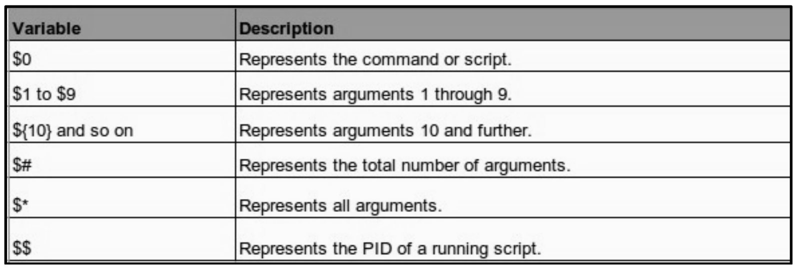
		`    echo "Even"`
		`  else`
		`    echo "Odd"`
		`  fi`
		`fi`

String-based Condition
The string-based condition returns a binary expression as a result meaning, it returns true if the specified condition is satisfied otherwise, it returns false.
	• == Returns true if the strings are equal
	• != Returns true if the strings are not equal
	• -n Returns true if the string to be tested is not null
	• -z Returns true if the string to be tested is null
		Script to illustrate string based conditions
		`read -p "First String: " str1`
		`read -p "Second String: " str2`
		`if [ -z "$str1" ]`
		`then`
		`  echo "The 1st string is null"`
		`elif [ -z "$str2" ]`
		`then`
		`  echo "The 2nd string is null"`
		`else`
		`  if [ $str1 == $str2 ]`
		`  then`
		`     echo "The strings are equal"`
		`  else`
		`    echo "The strings are not equal"`
		`  fi`
		`fi`

Command Line Arguments:

Command line arguments (also known as positional parameters) are the arguments specified at the command prompt with a command or script to be executed. The locations at the command prompt of the arguments as well as the location of the command, or the script itself, are stored in corresponding variables. These variables are special shell variables. Below picture will help you understand them.


Command arg1 arg2 arg3 arg4 arg5 arg6 arg7 arg8 arg9 arg10 ......
$0
$1
$2
$3
$4
$5
$6
$7
$8
$9
${10}
$#
$*



Variable	Description
$0	Represents the command or script.
$1 to $9	Represents arguments 1 through 9.
${10} and so on	Represents arguments 10 and further.
$#	Represents the total number of arguments.
$*	Represents all arguments.
$$	Represents the PID of a running script.

Within the command script, the passed parameters are accessible using ‘positional parameters’. These range from $0 to $9, where $0 refers to the name of the command itself, and $1 to $9 are the first through to the ninth parameter, depending on how many parameters were actually passed.

Example:

```
$ sh welcome fybca
```

Here $0 would be assigned sh

$1 would be assigned welcome

$2 would be assigned fybca

Example : Let’s create a shell script with name “command_line_agruments.sh”, it will show the command line argruments that were supplied and count number of agruments, value of first argument and Process ID (PID) of the Script.

```
echo “There are $# arguments specified at command line”
echo “the arguments supplied are : $*”
echo “the first argument is : $1”
echo “The  PID of script is:$$”
```

After running script , output will be

```
linuxvina@localhost:~$ ./command_line_agruments.sh Linux AIX HPUX VMware
There are 4 arguments specified at the command line.
The arguments supplied are: Linux AIX HPUX VMware
The first argument is: Linux
The PID of the script is: 16316
```

<u>Shifting Command Line Arguments</u>

The shift command is used to move command line arguments one position to the left. During this move, the first argument is lost. i.e. $2 will be shifted to $1 all the way to the tenth parameter being shifted to $9. The following “command_line_agruments.sh” script below uses the shift command:

```
    echo “There are $# arguments specified at command line”
    echo “the arguments supplied are : $*”
    echo “the first argument is : $1”
    echo “The  PID of script is:$$”
    shift
    echo “the new first argument after the first shift is :$1”
shift
echo “the new first argument after the second shift is :$1”
```

After running script , output will be

```
linuxtvina@localhost:~$ ./command_line_agruments.sh Linux AIX HPUX VMware
There are 4 arguments specified at the command line
The arguments supplied are: Linux AIX HPUX VMware
The first argument is: Linux
The Process ID of the script is: 16369
The new first argument after the first shift is: AIX
The new first argument after the second shift is: HPUX
```

Exercises

Set A:

1. Write a shell script to check whether two numbers are same or different.
2. Write a shell script to check whether given number is odd or even.
3. Write a shell script to display “ Good Morning”, “ Good afternoon” , and “Good Evening” depending on the hour .
4. Write a shell script to test a given file and return a message whether the file is a block device, a character device or a normal file.
5. Write a shell script that accepts file name as argument and converts all of them to uppercase, provided they exist in the current directory.
6. Write a shell script that accept directory name, if directory does not exist then it will create directory of same name.

Set B:

1. Write a shell script to accept argument string on command line , and display present working directory if argument string is “current” ,display parent directory if argument string is “parent” and display the contents of root directory if argument string is “root” .
2. Write a shell script to accept an extension name such as txt on command line and display the contents of all files with this extension, if there exists a file with this extension or give appropriate message.
3. Write a shell script to accept as argument an extension name such as .txt and move the contents of all files with this extension to a directory by the same name.
4. Write a shell script to accept a file name, and display file details if the file exists and a suitable message if it does not.
5. Write a shell script which receives two file names as arguments. It should check whether the two file contents are same or not. If they are same then second file should be deleted.
6. Write a shell script to check whether file is readable, writable or both or executable.

Set C:

1. Write a shell script that computes the gross salary of a employee according to the following rules:
   i) If basic salary is < 1500 then HRA =10% of the basic and DA =90% of the basic.
   ii) If basic salary is >=1500 then HRA =Rs500 and DA=98% of the basic

   The basic salary is entered interactively through the key board.
2. Write a shell script that accepts a file name, starting and ending line numbers as arguments and displays all the lines between the given line numbers.
3. Write a shell script that accept number and display whether that number is single digit, two digit or three digit.

4. Write menu driven program to perform arithmetic operations like +, - , *, /
5. Write menu driven program to perform the following tasks
   a) Show today's date and time
   b) Show files in current working directory.
   c) Show calendar
   d) Start editor to write letters
6. Write menu driven program to perform the following tasks
   a) Create directory
   b) Creating file
   c) Displaying contents of file
   d) Copying file into another file
   e) Displaying files in directory

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well done [ ]

Signature of the instructor: ______________________Date:______________________

Assignment 9

Aim: To study different looping control structures in shell programming.
Pre-requisite: Knowledge of conditional statements.

The student should read following topics before starting exercises.

Looping Control Structures:

Looping control structure, also known as Repetition control structure, is a type of control structure in programming languages that is used to simplify repetitive or recursive tasks. The following are looping control structures which can be used in shell programming

- while statement
- until statement
- for statement

To alter the flow of loop statements, two commands are used they are,

- break
- continue

looping Control structure	**Syntax**	**Example**

While statement executes an action as long as its test command is true.	while [condition]
	do
	Statement to be executed
	done
		Script to illustrate while loop which prints 1 to 10.
		i = 1
		while [$i -le 10]
		do
		echo $i
		i = \`expr $i + 1\`
		done

until statement
The until loop is useful when you need to execute a set of commands until a condition is true
	until [condition]
	do
	Statement to be executed
	done
		Script to illustrate until loop which prints 1 to 10.
		i=1
		until [!$i -lt 10]
		do
		echo $i
		i=\`expr $i + 1\`
		done

for - in statement
For - in is designed for use with lists of values; the variable operand is consecutively assigned the values in the list.
	for variable in list
	do
	commands
	done
		Script to illustrate use for loop
		for no in {1..10}
		do
		echo $no
		done

break statement
The break statement is used to terminate the execution of the entire loop, after completing the execution of all of the lines of code up to the break statement
	break	a=0
		
		while [$a -lt 10]
		do
		echo $a
		if [$a -eq 5]
		then
		  break
		fi
		a=`expr $a + 1`
		done

The continue statement is similar to the break command, except that it causes the current iteration of the loop to exit, rather than the entire loop.
 	
This statement is useful when an error has occurred but you want to try to execute the next iteration of the loop.
	continue	for a in 1 2 3 4 5 6 7 8 9 10
		do
		 if [$a == 5]
		 then
		    continue
		 fi
		 echo "Iteration no $a"
		done

Exercises

Set A:

1. Write a shell script to find factorial of number.
2. Write a shell script to find sum of digits.
3. Write a shell script to print multiplication table using command line arguments.
4. Write a shell script sum.sh that takes an unspecified number of command line arguments (up to 9) of int and finds their sum. Modify the code to add a number to the sum only if the number is greater than 10.
5. Write a shell script that accepts two integers as its arguments and computers the value of first number raised to the power of the second number.
6. Write a shell script to display first 10 odd numbers and their sum.

Set B:

1. Write a shell script which will print the numbers 1 - 10 (each on a separate line) and whether they are even or odd.
2. Write a shell script to display all the *.conf file that begins with either a, b, or, c or d under / etc directory.
3. Write a shell script to display days of week and adds “(WEEKEND)” to Sat and Sun, and “(weekday)” to rest of the days. Output should be following:

```
Day 1 : Mon (weekday)
----
Day 5 : Fri (weekday)
```

Day 6 : Sat (WEEKEND)
Day 7 : Sun (WEEKEND)

4. Write a shell script that will report the number of lines in each file within the current directory.
5. Write a shell script that accepts any number of arguments and prints them in a reverse order.
6. Write a shell script which will take a single command line argument (a directory) and will print each entry in that directory. If the entry is a file it will print its size. If the entry is a directory it will print how many items are in that directory.

Set C :

1. Write a shell script that takes a name of a folder as a command line argument, and produce a file that contains the names of all sub folders with size 0 (that is empty sub folders)
2. Write a shell script that takes a name of a folder, and delete all sub folders of size 0
3. Write a shell script that will take an input file and remove identical lines (or duplicate lines from the file)
4. Write a shell script that deletes all lines containing a specified word in one or more files supplied as arguments to it.
5. Write a shell script which accepts a filename, displays menu with following options, accepts user choice as number and takes appropriate action
   a) Display the file contents
   b) Display the file Size in blocks
   c) Display the number of words in file
   d) Display last five lines of the file
   e) Display first ten lines of the file

6. Write a shell script that displays menu with following options, accepts user choice as number and takes appropriate actions
   a) Displays the No of users logged in
   b) Display the login id of user logged i
   c) Display the present working directory
   d) Display the home directory of logged in user
   e) Display the path

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well done [ ]

Signature of the instructor: __________________________ Date: ________________________