Practical File
Of
Operating System
22CS005

Submitted

in partial fulfillment for the award of the degree

of

BACHELEOR OF ENGINEERING

in

COMPUTER SCIENCE & ENGINEERING

CHITKARA UNIVERSITY

CHANDIGARH-PATIALA NATIONAL HIGHWAY
RAJPURA (PATIALA) PUNJAB-140401 (INDIA)

June, 2024

Submitted To:	**Submitted By:**
Dr. Meenakshi Malhotra	Jay Soni
Associate Professor	2310991943
Chitkara University, Punjab	G-22,Sem-2

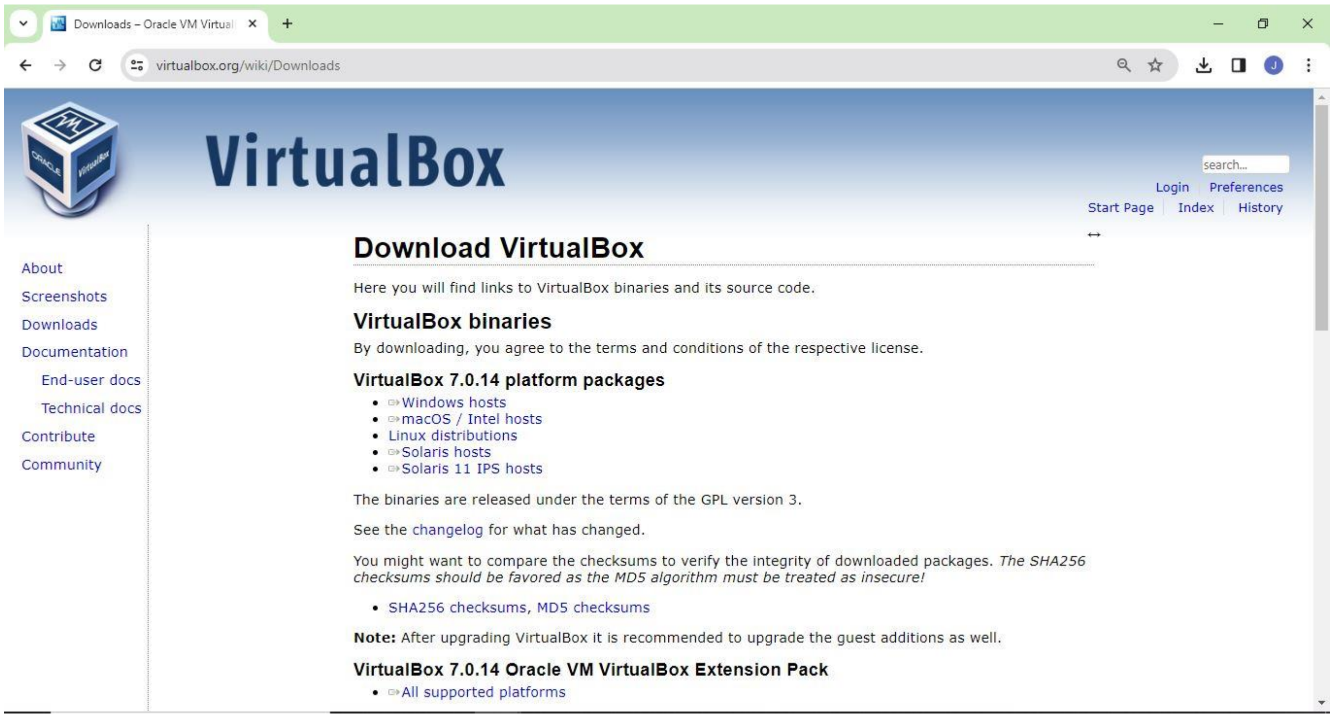
INDEX

Program 1: a) Installation: Configuration & Customizations of Linux

LINUX: Linux is an open-source Unix-like operating system-based family on the Linux kernel, and the OS kernel was first published on 17 September 1991 by Linus Torvalds. Typically, Linux is packaged as the Linux distribution, which contains the supporting libraries, system software, and kernel, several of which are offered by the GNU Project. Linux powers servers, desktops, smartphones, and embedded systems, offering various customization options.

Virtual Box: VirtualBox is an open-source software for virtualization that allows users to run multiple operating systems on a single machine simultaneously. Essentially, with VirtualBox, you can emulate a particular computer system on your device without affecting your primary OS.

Installation of Virtual Box:
To install the virtual box, first go to your web browser and type https://www.virtualbox.org/wiki/Downloads.
When you go to this link you will be able to see various platform packages.



Download VirtualBox

Here you will find links to VirtualBox binaries and its source code.

VirtualBox binaries

By downloading, you agree to the terms and conditions of the respective license.

VirtualBox 7.0.14 platform packages

- Windows hosts
- macOS / Intel hosts
- Linux distributions
- Solaris hosts
- Solaris 11 IPS hosts

The binaries are released under the terms of the GPL version 3.

See the changelog for what has changed.

You might want to compare the checksums to verify the integrity of downloaded packages. *The SHA256 checksums should be favored as the MD5 algorithm must be treated as insecure!*

- SHA256 checksums, MD5 checksums

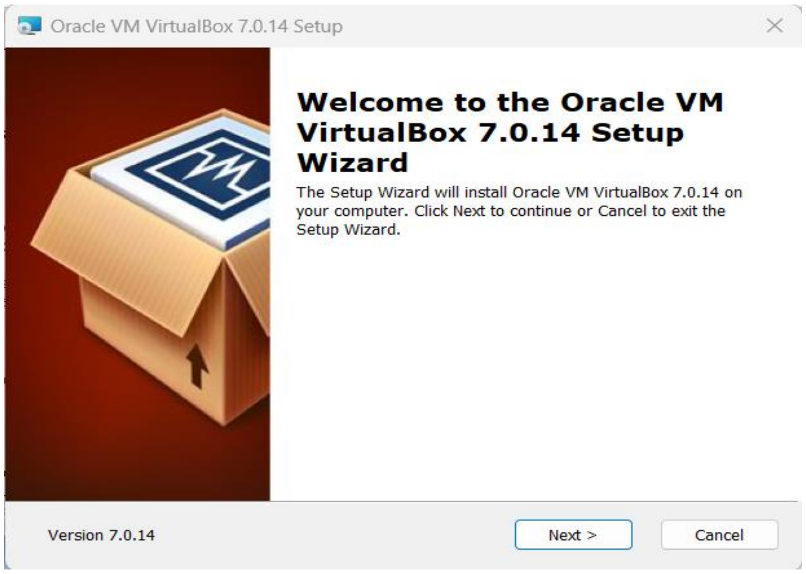
Note: After upgrading VirtualBox it is recommended to upgrade the guest additions as well.

VirtualBox 7.0.14 Oracle VM VirtualBox Extension Pack

- All supported platforms

From the various options, we are going to select Windows Hosts and when we click on it our virtual box will start getting installed.

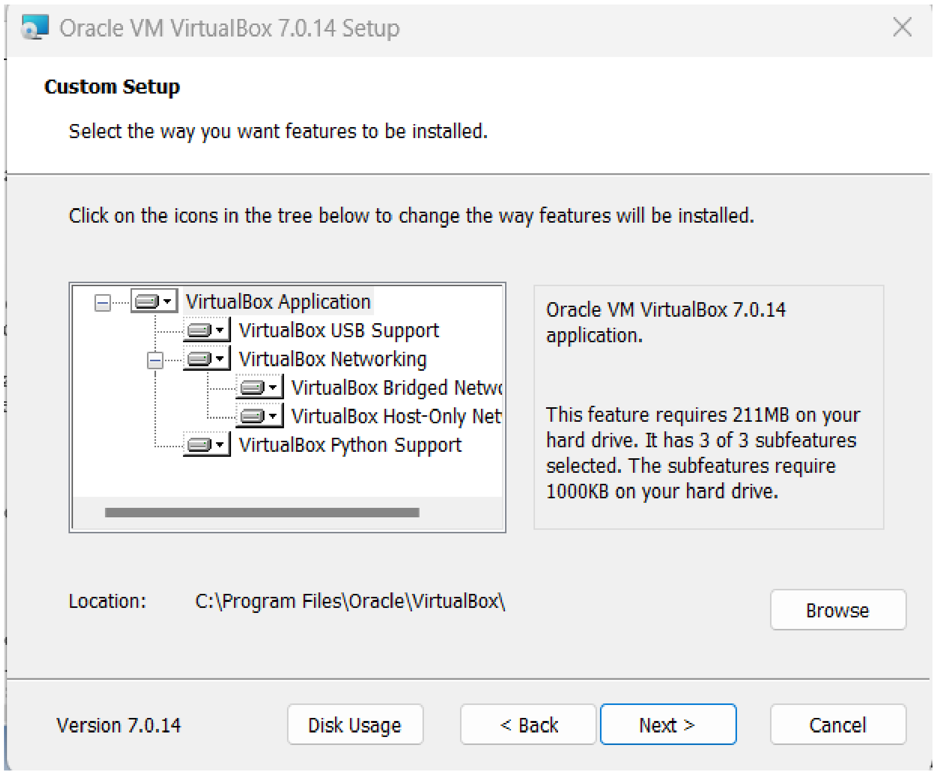
Once the download is complete, open setup file and follow the steps below:
Step 1-Click on next.



Welcome to the Oracle VM VirtualBox 7.0.14 Setup Wizard

The Setup Wizard will install Oracle VM VirtualBox 7.0.14 on your computer. Click Next to continue or Cancel to exit the Setup Wizard.

Step 2- Select the location where you want to install the virtual box and click on next.



Custom Setup

Select the way you want features to be installed.

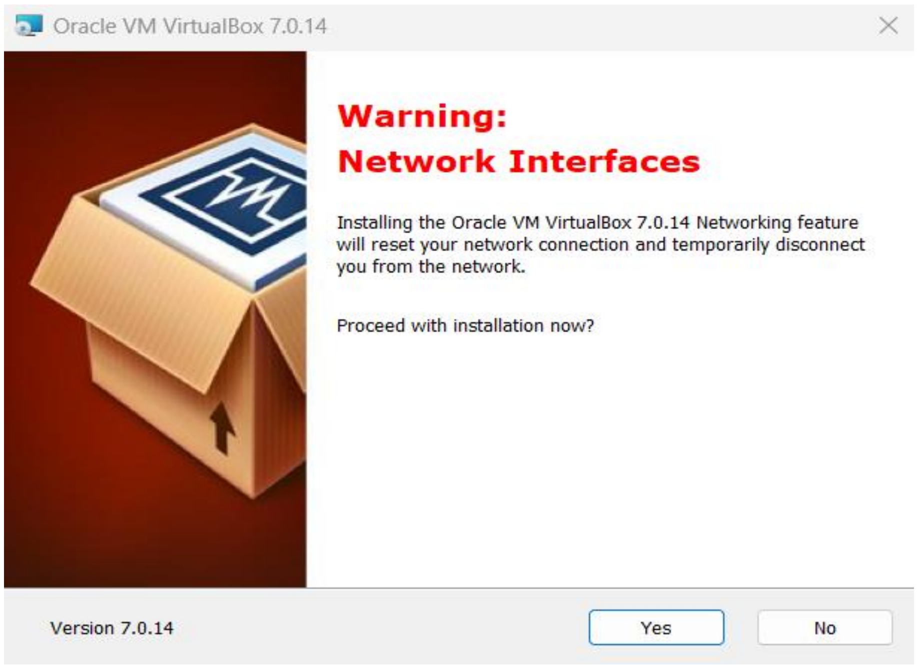
Click on the icons in the tree below to change the way features will be installed.

Oracle VM VirtualBox 7.0.14 application.

This feature requires 211MB on your hard drive. It has 3 of 3 subfeatures selected. The subfeatures require 1000KB on your hard drive.

Location: C:\Program Files\Oracle\VirtualBox\

Step 3-Now click on yes.

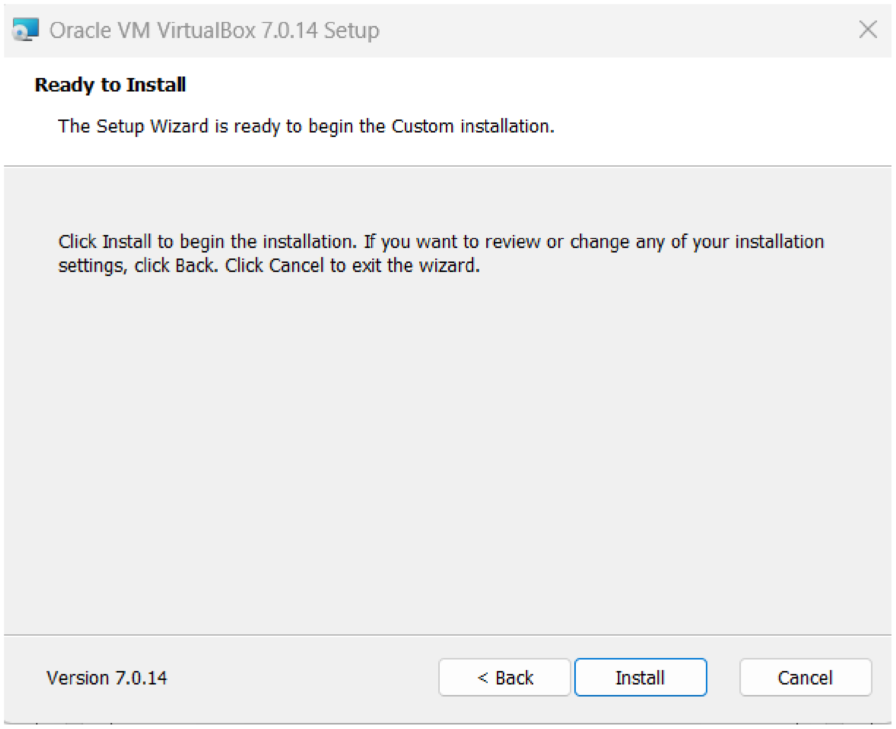


Warning:
Network Interfaces

Installing the Oracle VM VirtualBox 7.0.14 Networking feature will reset your network connection and temporarily disconnect you from the network.

Proceed with installation now?

Step 4- Click on Install to install Linux on Windows.

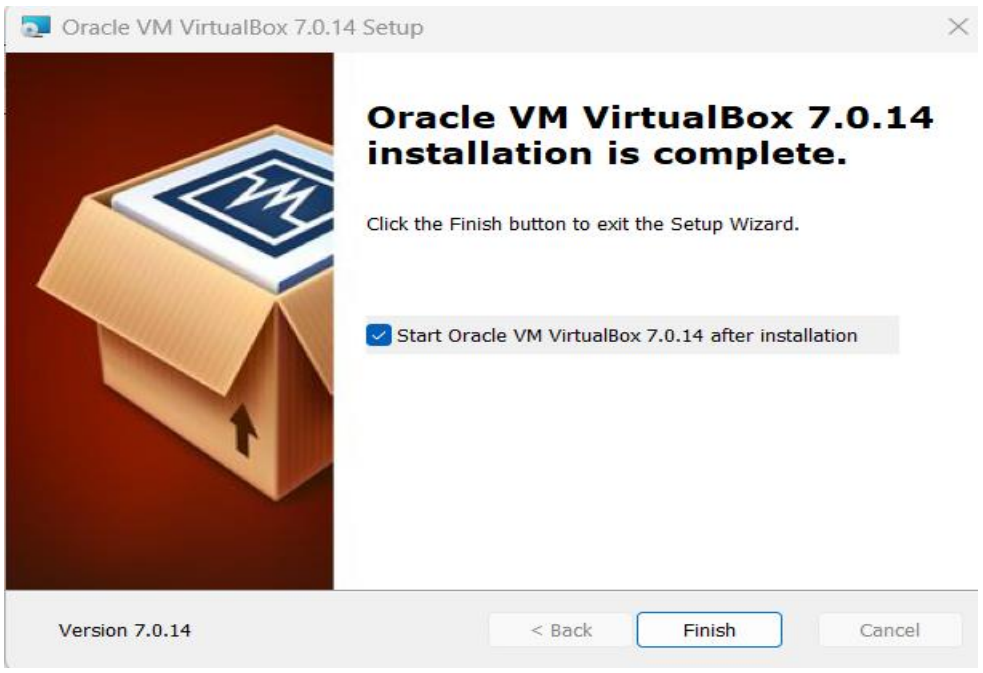


Ready to Install

The Setup Wizard is ready to begin the Custom installation.

Click Install to begin the installation. If you want to review or change any of your installation settings, click Back. Click Cancel to exit the wizard.

Step 5-Now installation of virtual box will start. Once complete, click on the Finish Button to start Virtual box.

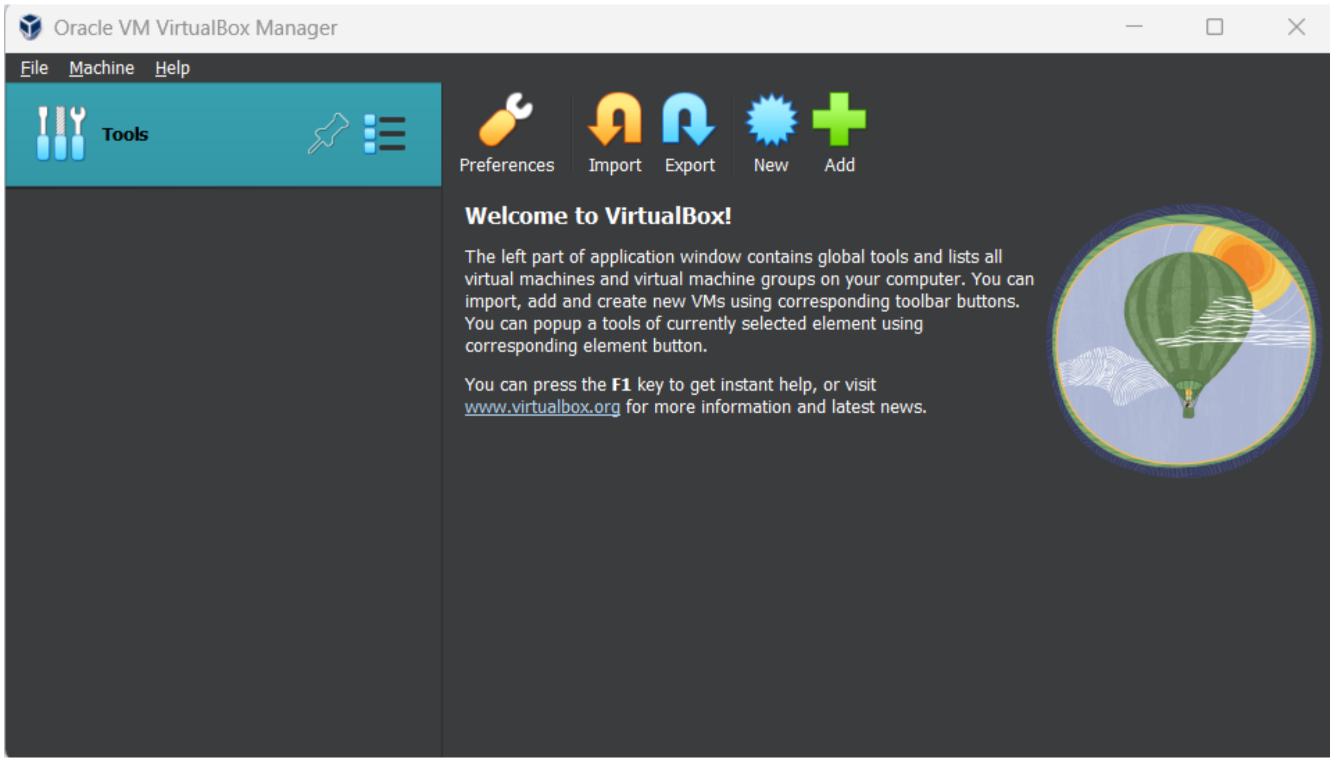


Oracle VM VirtualBox 7.0.14 installation is complete.

Click the Finish button to exit the Setup Wizard.

[✓] Start Oracle VM VirtualBox 7.0.14 after installation

The virtual box dashboard looks like this.



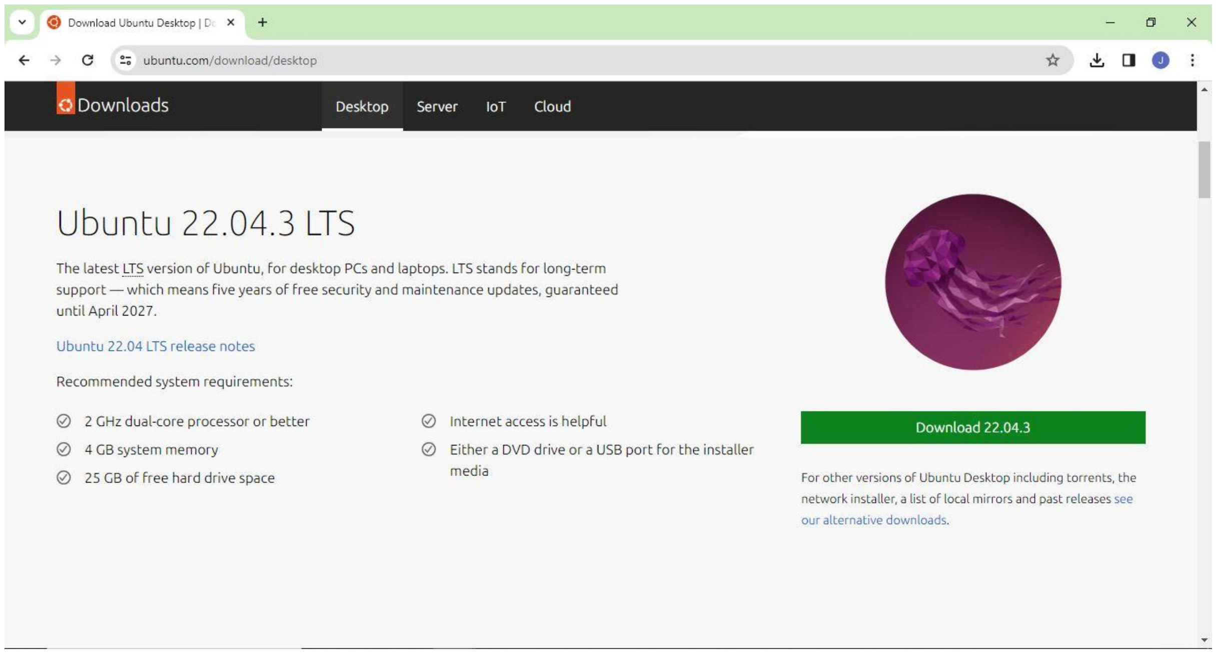
Welcome to VirtualBox!

The left part of application window contains global tools and lists all virtual machines and virtual machine groups on your computer. You can import, add and create new VMs using corresponding toolbar buttons. You can popup a tools of currently selected element using corresponding element button.

You can press the **F1** key to get instant help, or visit www.virtualbox.org for more information and latest news.

Download Ubuntu:

To download Ubuntu visit the link https://ubuntu.com/download/desktop



Ubuntu 22.04.3 LTS

The latest LTS version of Ubuntu, for desktop PCs and laptops. LTS stands for long-term support — which means five years of free security and maintenance updates, guaranteed until April 2027.

Ubuntu 22.04 LTS release notes

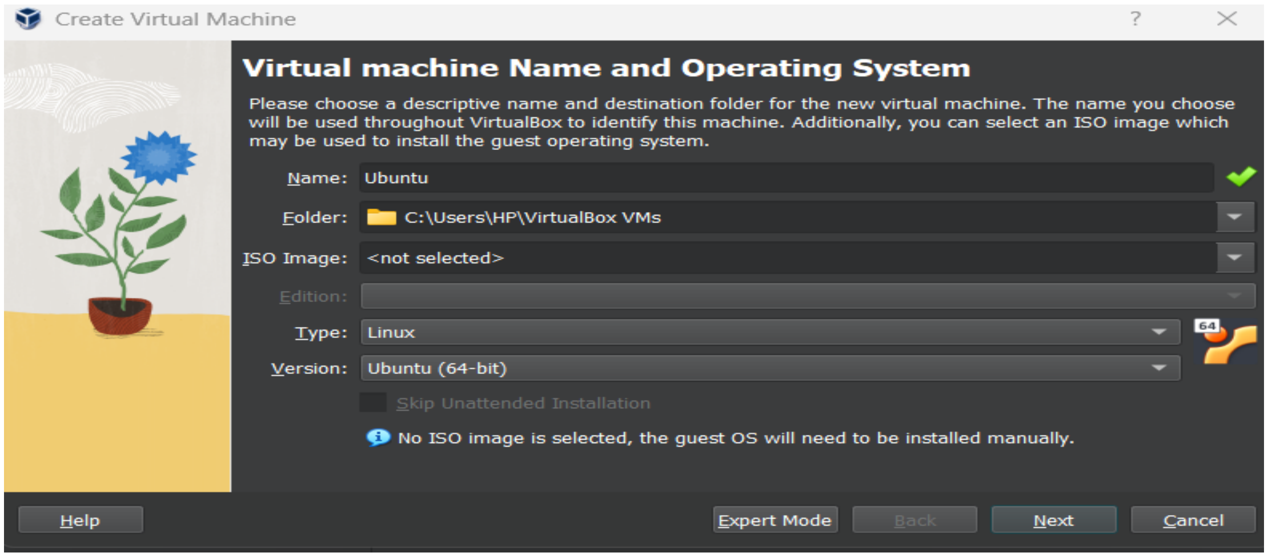
Recommended system requirements:

- 2 GHz dual-core processor or better
- 4 GB system memory
- 25 GB of free hard drive space
- Internet access is helpful
- Either a DVD drive or a USB port for the installer media

For other versions of Ubuntu Desktop including torrents, the network installer, a list of local mirrors and past releases see our alternative downloads.

Create a machine in virtual box:

Step 1-Open virtual box and click on the new button. A new window will appear and give the name of the OS you are installing and select OS as Linux and version as Ubuntu(64 bit). Click on next.



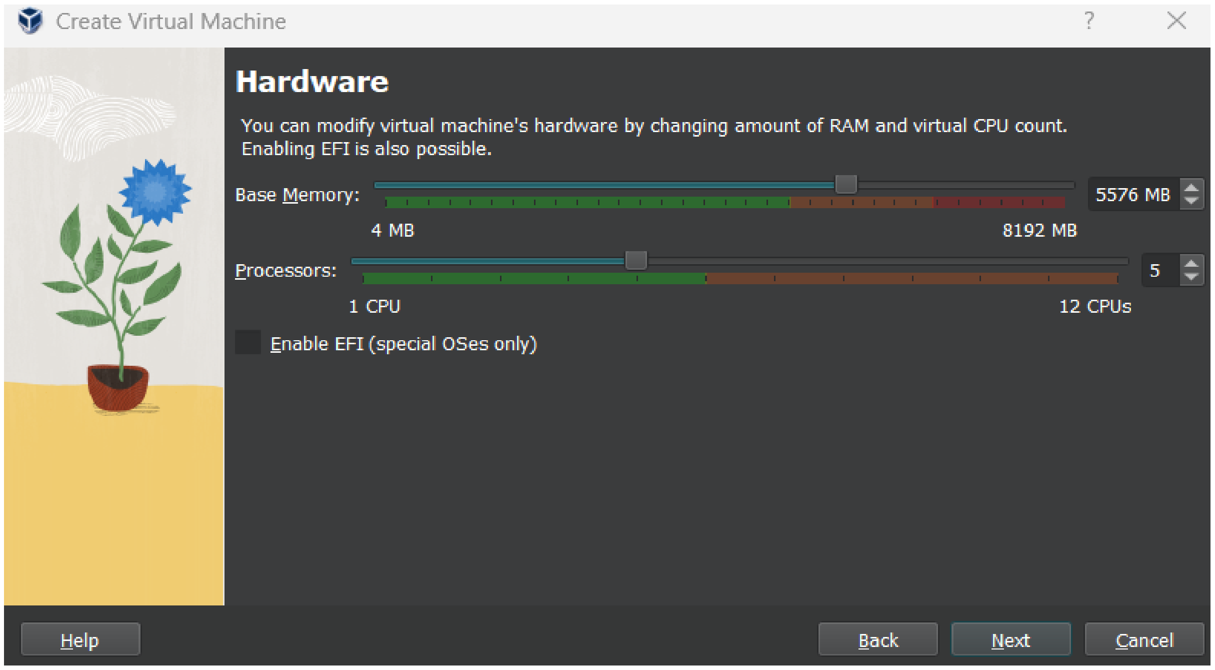
Virtual machine Name and Operating System

Please choose a descriptive name and destination folder for the new virtual machine. The name you choose will be used throughout VirtualBox to identify this machine. Additionally, you can select an ISO image which may be used to install the guest operating system.

Name: Ubuntu
Folder: C:\Users\HP\VirtualBox VMs
ISO Image: <not selected>
Edition:
Type: Linux
Version: Ubuntu (64-bit)
Skip Unattended Installation

No ISO image is selected, the guest OS will need to be installed manually.

Step 2-Now allocate Base memory and processors to your virtual os.



Hardware

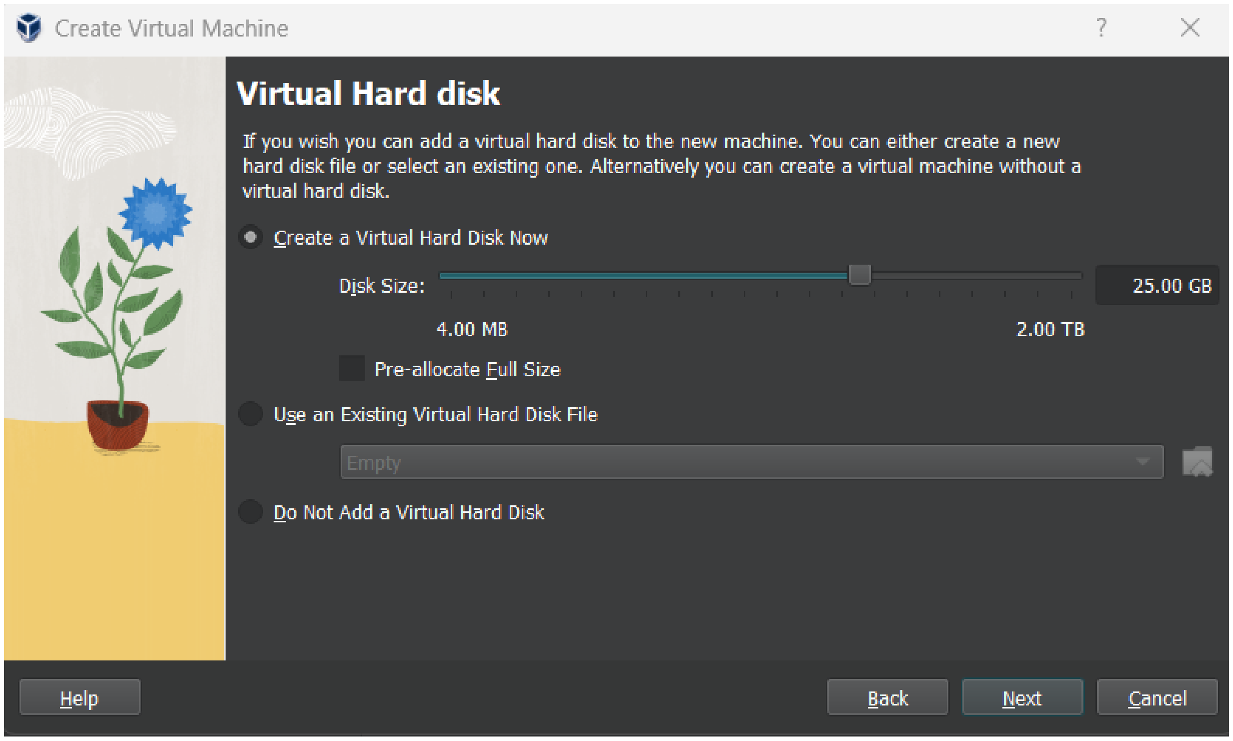
You can modify virtual machine's hardware by changing amount of RAM and virtual CPU count. Enabling EFI is also possible.

Base Memory: 5576 MB (4 MB – 8192 MB)

Processors: 5 (1 CPU – 12 CPUs)

Enable EFI (special OSes only)

Step 3-Now create virtual hard disk of any size you want.



Virtual Hard disk

If you wish you can add a virtual hard disk to the new machine. You can either create a new hard disk file or select an existing one. Alternatively you can create a virtual machine without a virtual hard disk.

[✓ Create a Virtual Hard Disk Now]

Disk Size: 25.00 GB (4.00 MB – 2.00 TB)

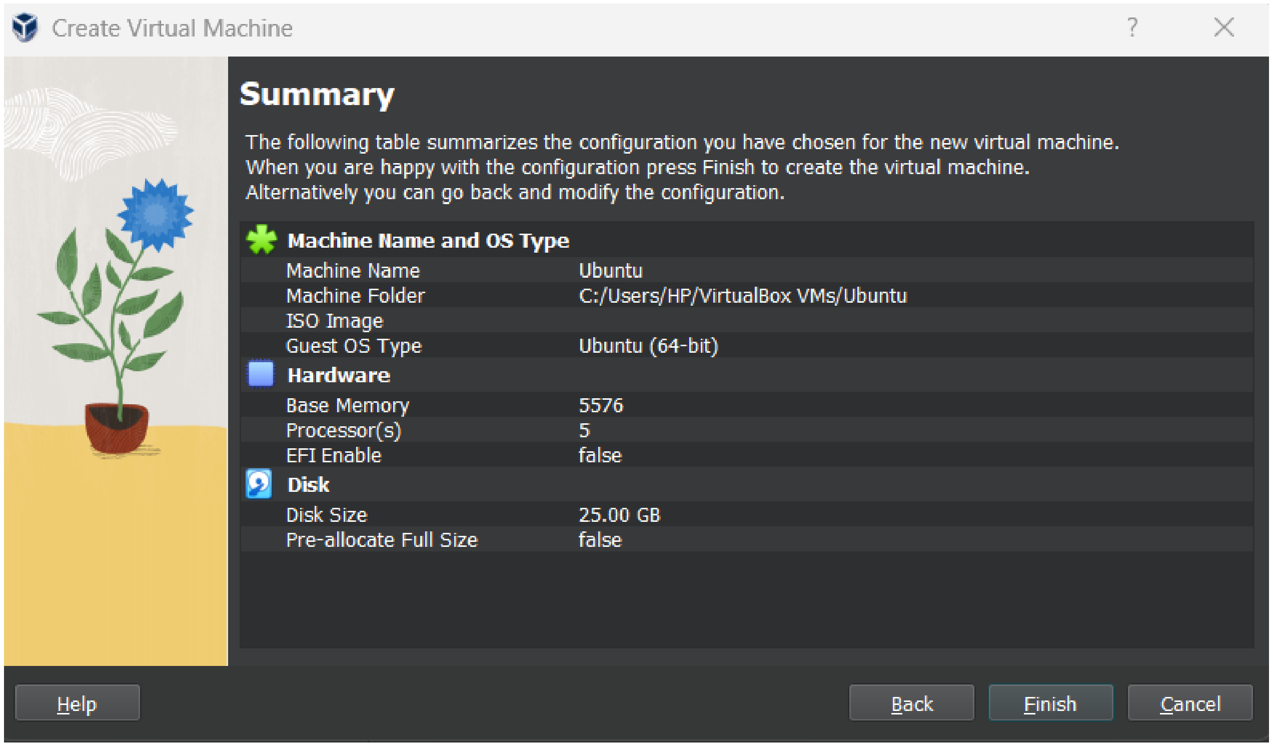
Pre-allocate Full Size

Use an Existing Virtual Hard Disk File

Empty

Do Not Add a Virtual Hard Disk

Step 4-Now click on finish.

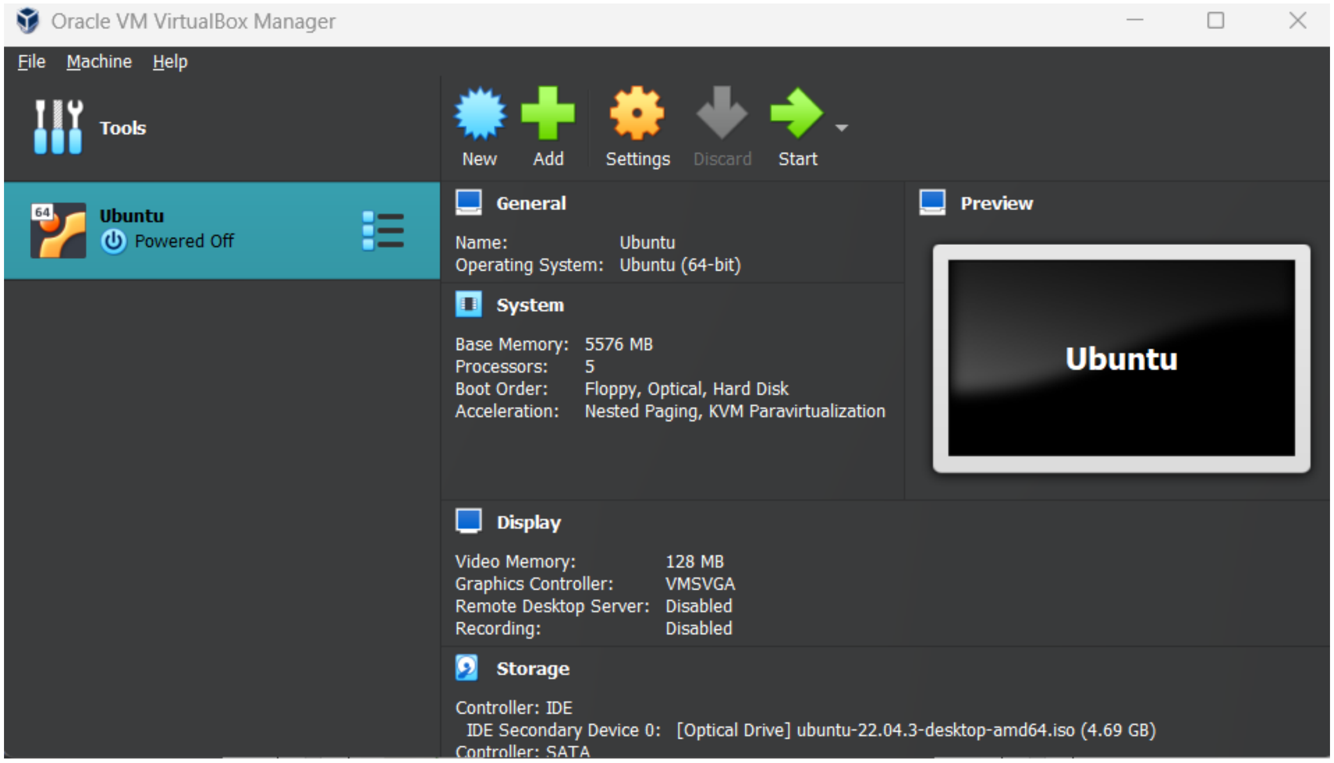


Summary

The following table summarizes the configuration you have chosen for the new virtual machine. When you are happy with the configuration press Finish to create the virtual machine. Alternatively you can go back and modify the configuration.

Machine Name and OS Type	
Machine Name	Ubuntu
Machine Folder	C:/Users/HP/VirtualBox VMs/Ubuntu
ISO Image	
Guest OS Type	Ubuntu (64-bit)
Hardware	
Base Memory	5576
Processor(s)	5
EFI Enable	false
Disk	
Disk Size	25.00 GB
Pre-allocate Full Size	false

Step 5-Now you can see the machine name in left panel.



General

Name: Ubuntu
Operating System: Ubuntu (64-bit)

System

Base Memory: 5576 MB
Processors: 5
Boot Order: Floppy, Optical, Hard Disk
Acceleration: Nested Paging, KVM Paravirtualization

Display

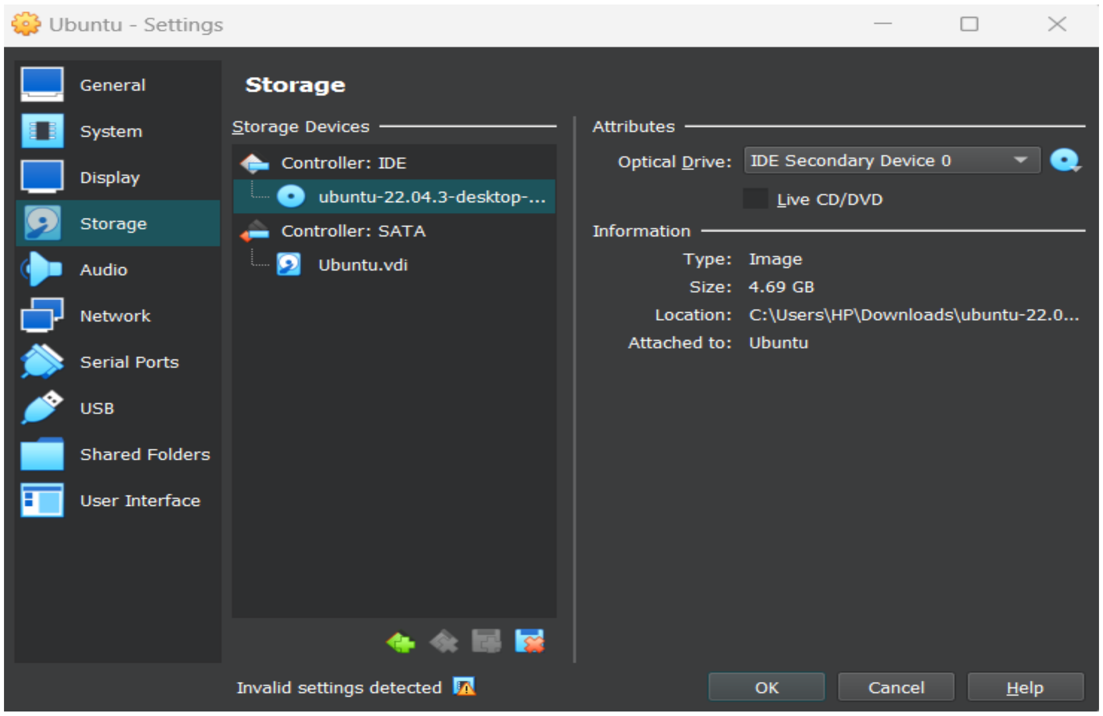
Video Memory: 128 MB
Graphics Controller: VMSVGA
Remote Desktop Server: Disabled
Recording: Disabled

Storage

Controller: IDE
IDE Secondary Device 0: [Optical Drive] ubuntu-22.04.3-desktop-amd64.iso (4.69 GB)
Controller: SATA

How to install Ubuntu

Step 1-Go to settings and then click on storage and add the .iso file of the ubuntu.



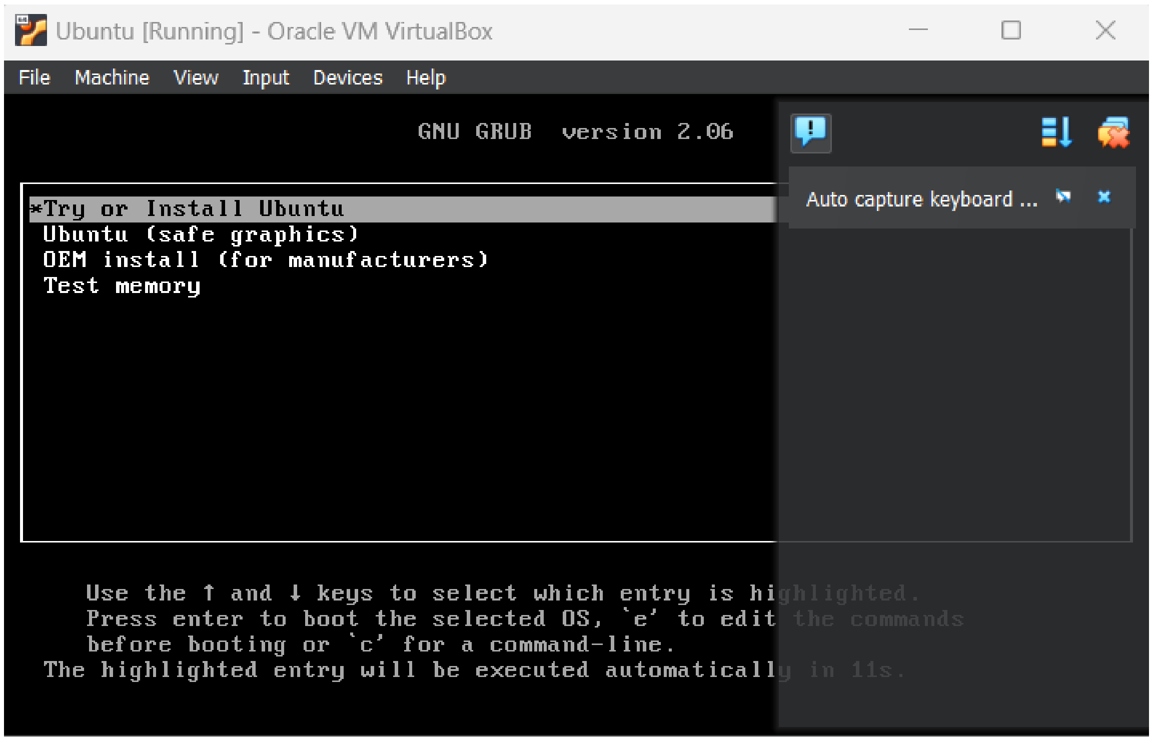
Controller: IDE
ubuntu-22.04.3-desktop-...
Controller: SATA
Ubuntu.vdi

Optical Drive: IDE Secondary Device 0
Live CD/DVD

Information
Type: Image
Size: 4.69 GB
Location: C:\Users\HP\Downloads\ubuntu-22.0...
Attached to: Ubuntu

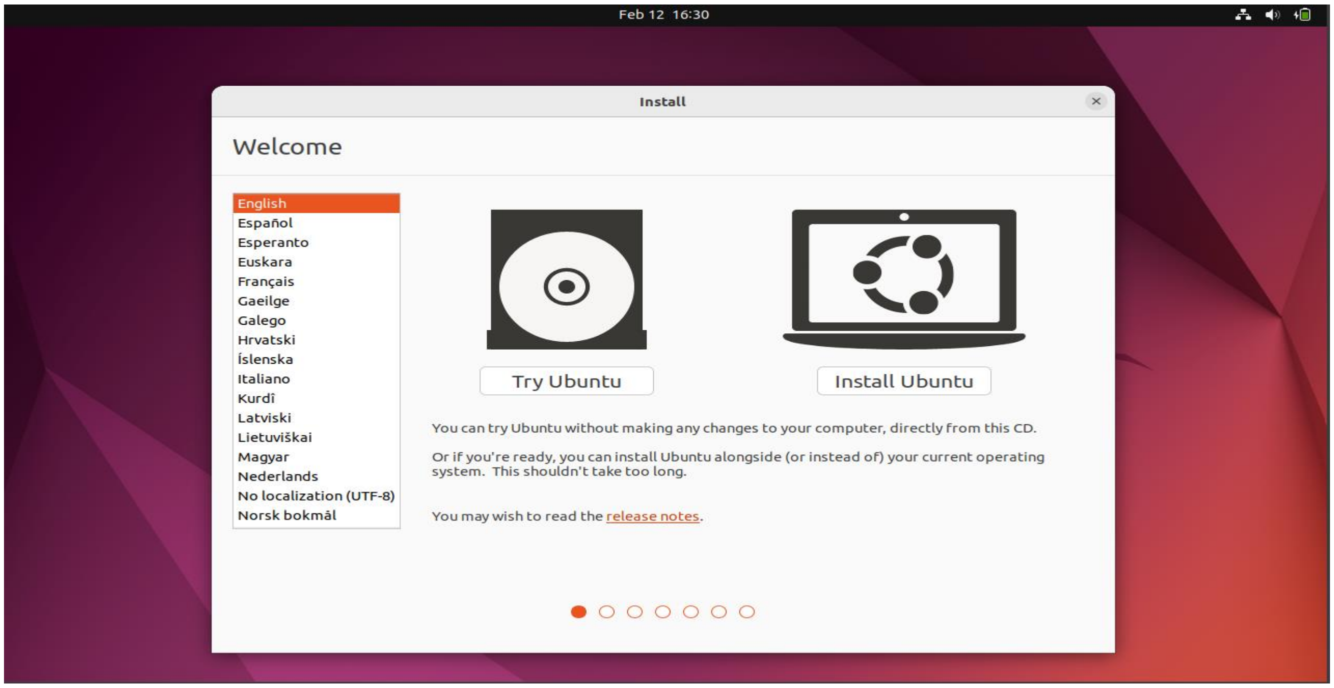
Invalid settings detected

Step 2-Click on Start and select try and install Ubuntu.


Ubuntu [Running] - Oracle VM VirtualBox
File
Machine
View
Input
Devices
Help
Auto capture keyboard ...

```
GNU GRUB  version 2.06

*Try or Install Ubuntu
 Ubuntu (safe graphics)
 OEM install (for manufacturers)
 Test memory

   Use the ↑ and ↓ keys to select which entry is highlighted.
   Press enter to boot the selected OS, `e' to edit the commands
   before booting or `c' for a command-line.
The highlighted entry will be executed automatically in 11s.
```

Step 3-Now click on install ubuntu.

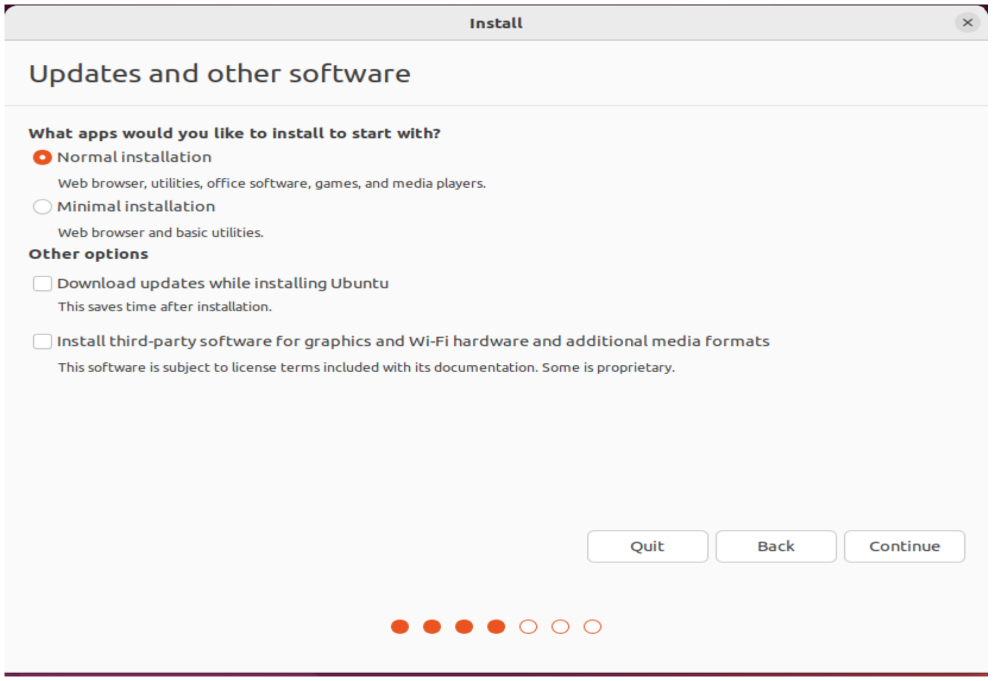


You can try Ubuntu without making any changes to your computer, directly from this CD.

Or if you're ready, you can install Ubuntu alongside (or instead of) your current operating system. This shouldn't take too long.

You may wish to read the release notes.

Step 4-Now click on continue



Updates and other software

What apps would you like to install to start with?

[✓ Normal installation]
Web browser, utilities, office software, games, and media players.

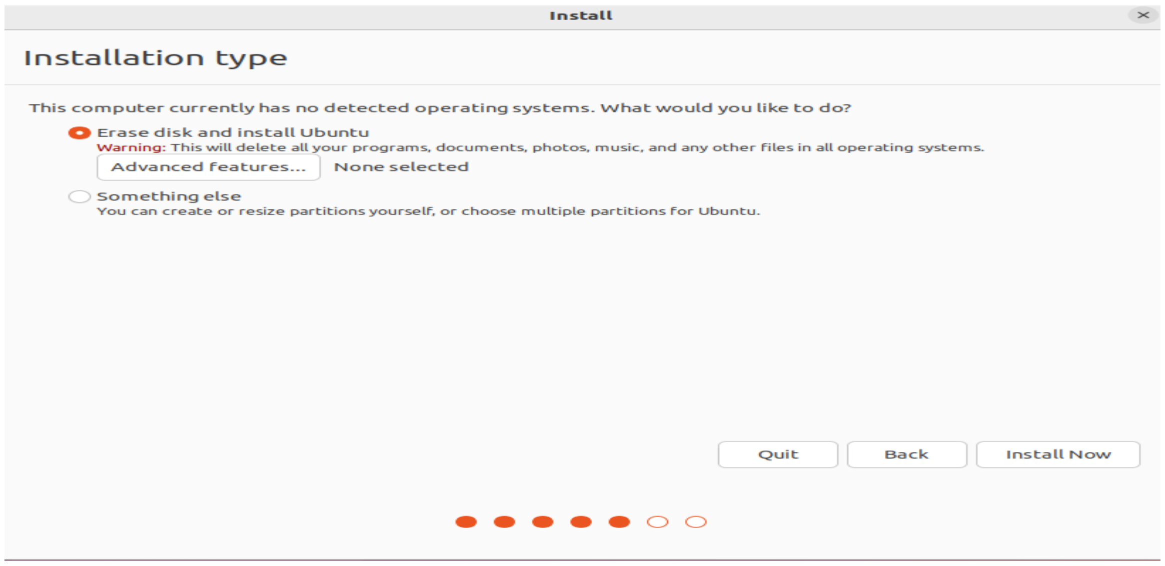
Minimal installation
Web browser and basic utilities.

Other options

Download updates while installing Ubuntu
This saves time after installation.

Install third-party software for graphics and Wi-Fi hardware and additional media formats
This software is subject to license terms included with its documentation. Some is proprietary.

Step 5-Select option to erase the disk and install ubuntu and click on install now.This option installs ubuntu into your virtual hard drive which is we made earlier.It will not harm your pc on windows installation.



Install

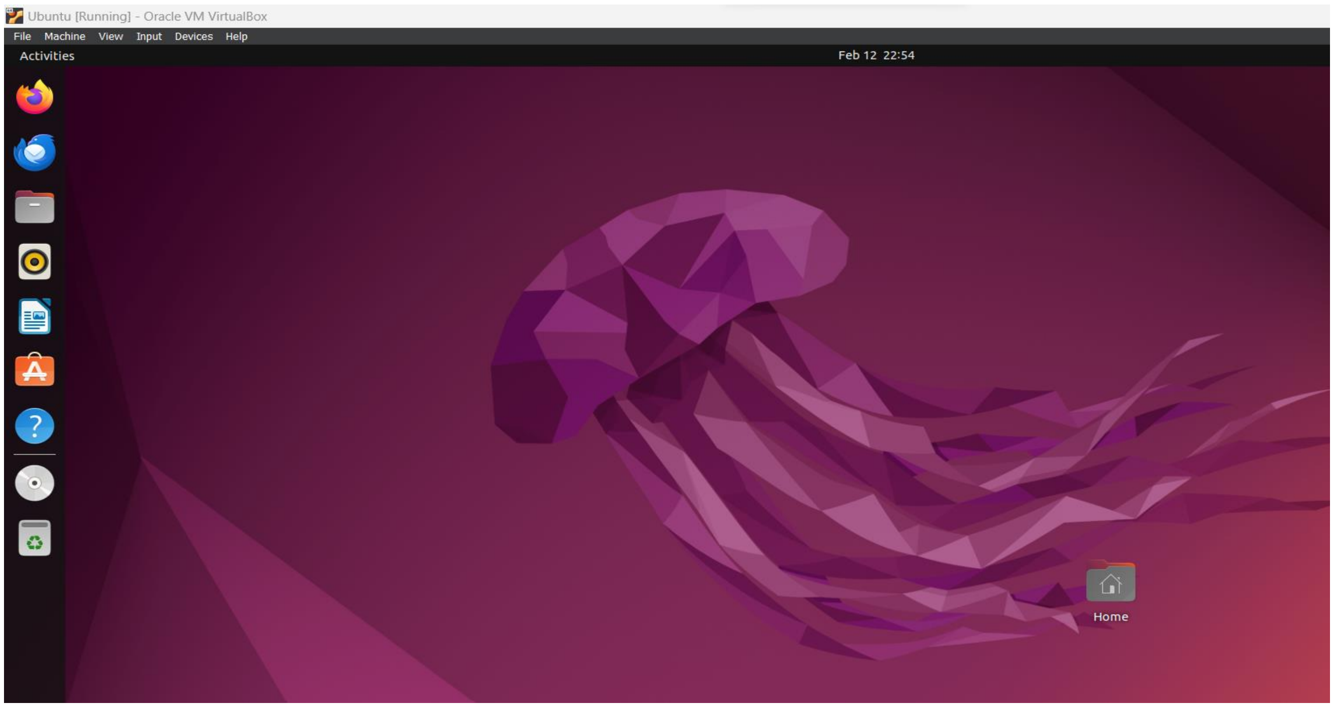
Installation type

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.
Advanced features... None selected

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

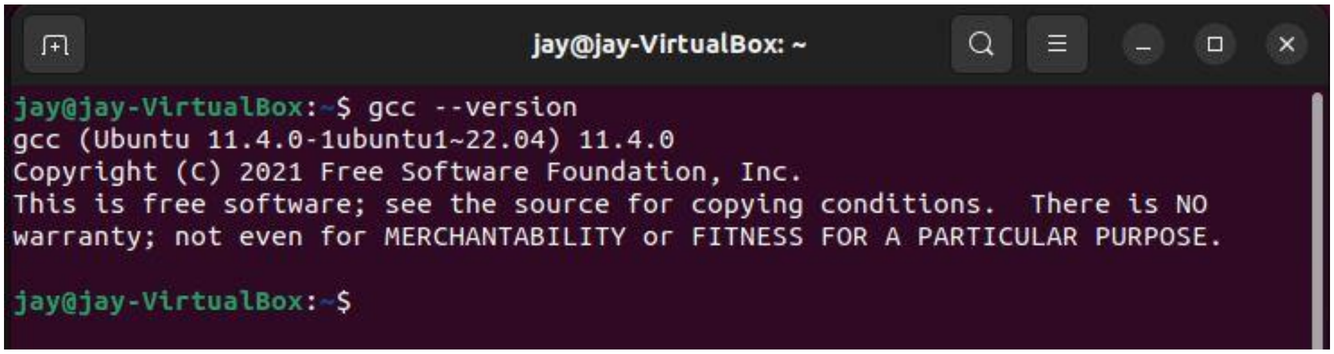
Step-6-Select your username and password for your Ubuntu admin account.
Step-7-Now the installation process starts.
After finishing the installation, you will see the Ubuntu desktop.


Ubuntu [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Feb 12 22:54
Home

Program 1(b): Introduction to GCC compiler: Basics of GCC, Compilation of program, Execution of program.

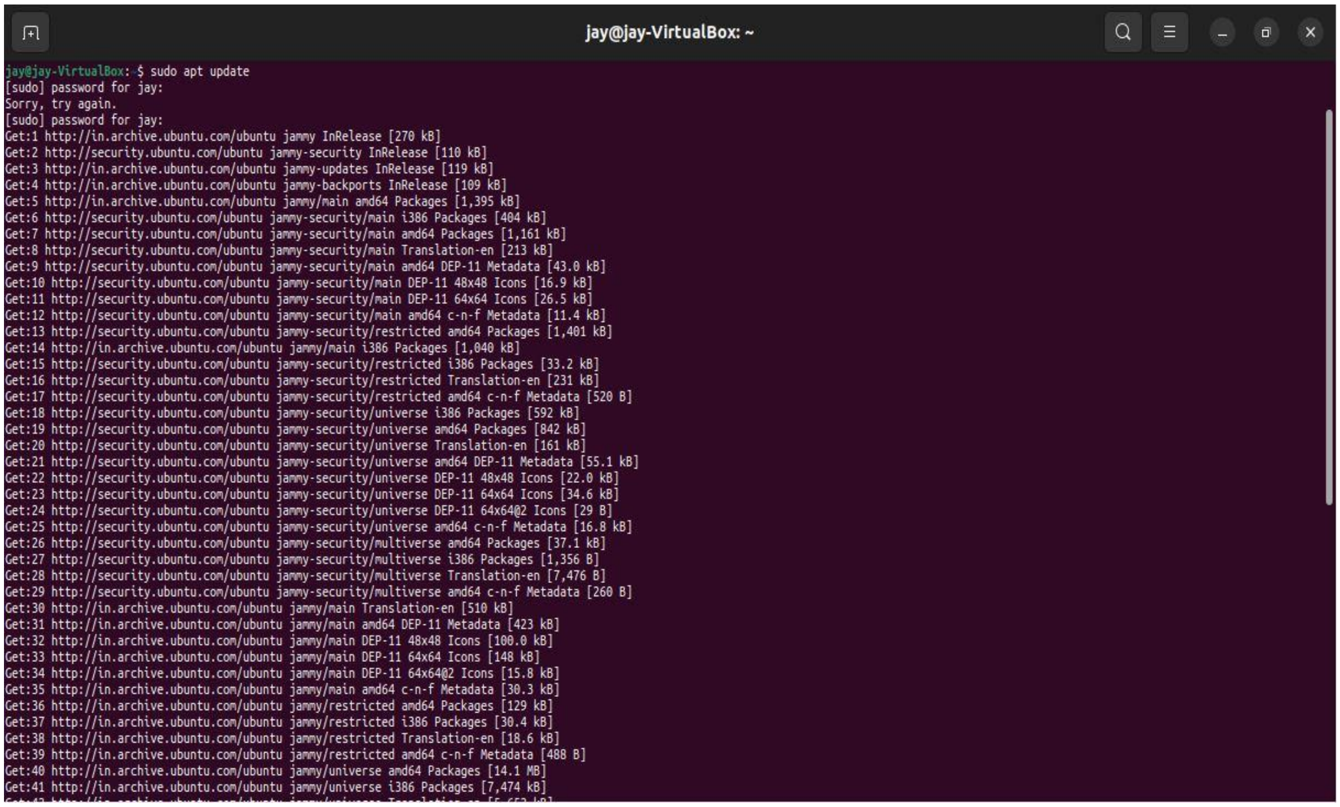
GCC compiler:
In Linux, the GCC stands for GNU Compiler Collection. It is a compiler system for the various programming languages. It is mainly used to compile the C and C++ programs. It takes the name of the source program as a necessary argument; rest arguments are optional such as debugging, warning, object file, and linking libraries.

Installation of GCC:
By default, it comes with the most Linux distributions. We can verify it by executing the below command:
$gcc -version
The above command will display the installed version of the GCC tool.



```
jay@jay-VirtualBox:~$ gcc --version
gcc (Ubuntu 11.4.0-1ubuntu1~22.04) 11.4.0
Copyright (C) 2021 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.

jay@jay-VirtualBox:~$
```

If it is not installed, follow the below steps to install it:
Step 1: Update the package list.
To update the package list, execute the following command:
$sudo apt update
It will ask for the system administrative password, enter the password. It will start updating the system package. Consider the snap of output on next page.


jay@jay-VirtualBox: ~

```
jay@jay-VirtualBox:~$ sudo apt update
[sudo] password for jay:
Sorry, try again.
[sudo] password for jay:
Get:1 http://in.archive.ubuntu.com/ubuntu jammy InRelease [270 kB]
Get:2 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:3 http://in.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:4 http://in.archive.ubuntu.com/ubuntu jammy-backports InRelease [109 kB]
Get:5 http://in.archive.ubuntu.com/ubuntu jammy/main amd64 Packages [1,395 kB]
Get:6 http://security.ubuntu.com/ubuntu jammy-security/main i386 Packages [404 kB]
Get:7 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [1,161 kB]
Get:8 http://security.ubuntu.com/ubuntu jammy-security/main Translation-en [213 kB]
Get:9 http://security.ubuntu.com/ubuntu jammy-security/main amd64 DEP-11 Metadata [43.0 kB]
Get:10 http://security.ubuntu.com/ubuntu jammy-security/main DEP-11 48x48 Icons [16.9 kB]
Get:11 http://security.ubuntu.com/ubuntu jammy-security/main DEP-11 64x64 Icons [26.5 kB]
Get:12 http://security.ubuntu.com/ubuntu jammy-security/main amd64 c-n-f Metadata [11.4 kB]
Get:13 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 Packages [1,401 kB]
Get:14 http://in.archive.ubuntu.com/ubuntu jammy/main i386 Packages [1,040 kB]
Get:15 http://security.ubuntu.com/ubuntu jammy-security/restricted i386 Packages [33.2 kB]
Get:16 http://security.ubuntu.com/ubuntu jammy-security/restricted Translation-en [231 kB]
Get:17 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 c-n-f Metadata [520 B]
Get:18 http://security.ubuntu.com/ubuntu jammy-security/universe i386 Packages [592 kB]
Get:19 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 Packages [842 kB]
Get:20 http://security.ubuntu.com/ubuntu jammy-security/universe Translation-en [161 kB]
Get:21 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 DEP-11 Metadata [55.1 kB]
Get:22 http://security.ubuntu.com/ubuntu jammy-security/universe DEP-11 48x48 Icons [22.0 kB]
Get:23 http://security.ubuntu.com/ubuntu jammy-security/universe DEP-11 64x64 Icons [34.6 kB]
Get:24 http://security.ubuntu.com/ubuntu jammy-security/universe DEP-11 64x64@2 Icons [29 B]
Get:25 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 c-n-f Metadata [16.8 kB]
Get:26 http://security.ubuntu.com/ubuntu jammy-security/multiverse amd64 Packages [37.1 kB]
Get:27 http://security.ubuntu.com/ubuntu jammy-security/multiverse i386 Packages [1,356 B]
Get:28 http://security.ubuntu.com/ubuntu jammy-security/multiverse Translation-en [7,476 B]
Get:29 http://security.ubuntu.com/ubuntu jammy-security/multiverse amd64 c-n-f Metadata [260 B]
Get:30 http://in.archive.ubuntu.com/ubuntu jammy/main Translation-en [510 kB]
Get:31 http://in.archive.ubuntu.com/ubuntu jammy/main amd64 DEP-11 Metadata [423 kB]
Get:32 http://in.archive.ubuntu.com/ubuntu jammy/main DEP-11 48x48 Icons [100.0 kB]
Get:33 http://in.archive.ubuntu.com/ubuntu jammy/main DEP-11 64x64 Icons [148 kB]
Get:34 http://in.archive.ubuntu.com/ubuntu jammy/main DEP-11 64x64@2 Icons [15.8 kB]
Get:35 http://in.archive.ubuntu.com/ubuntu jammy/main amd64 c-n-f Metadata [30.3 kB]
Get:36 http://in.archive.ubuntu.com/ubuntu jammy/restricted amd64 Packages [129 kB]
Get:37 http://in.archive.ubuntu.com/ubuntu jammy/restricted i386 Packages [30.4 kB]
Get:38 http://in.archive.ubuntu.com/ubuntu jammy/restricted Translation-en [18.6 kB]
Get:39 http://in.archive.ubuntu.com/ubuntu jammy/restricted amd64 c-n-f Metadata [488 B]
Get:40 http://in.archive.ubuntu.com/ubuntu jammy/universe amd64 Packages [14.1 MB]
Get:41 http://in.archive.ubuntu.com/ubuntu jammy/universe i386 Packages [7,474 kB]
```

Step 2: Install the build-essential package.

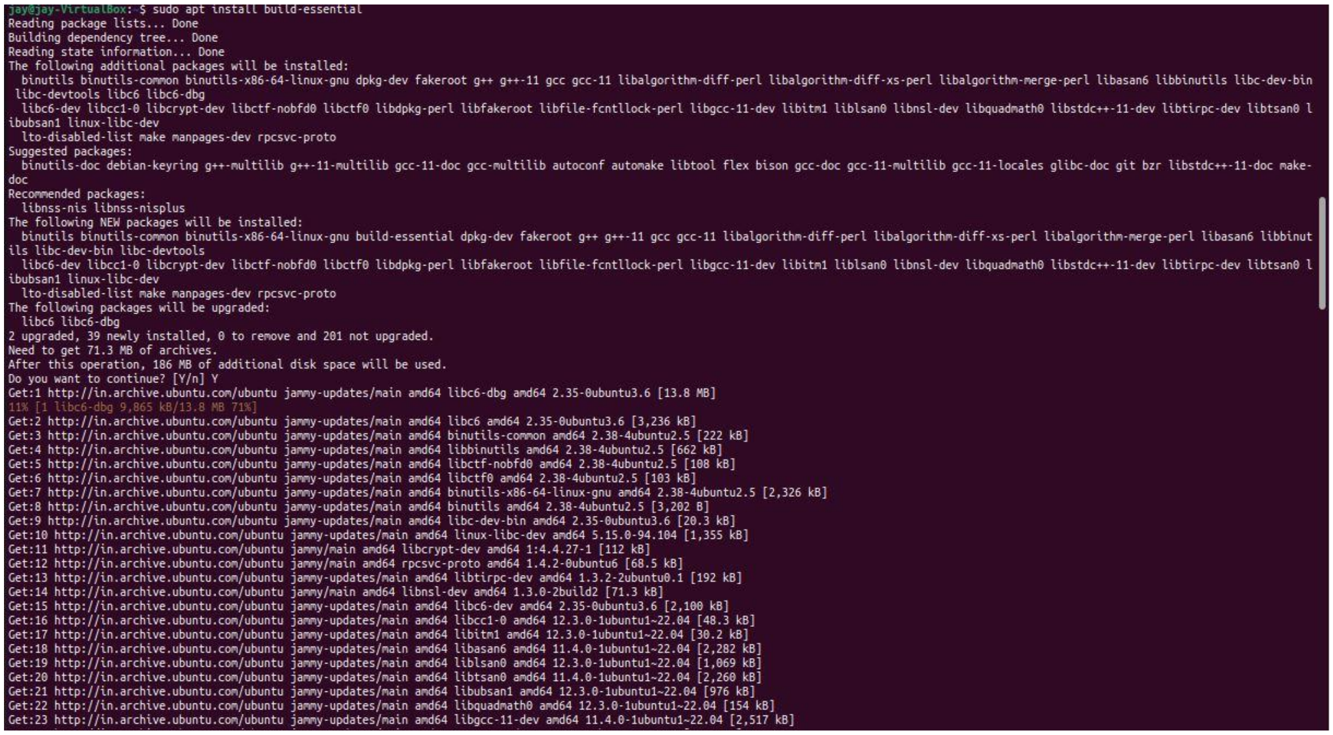
It contains various packages such as gcc, g++, and make utility.

Execute the below command to install it:

$sudo apt install build-essential

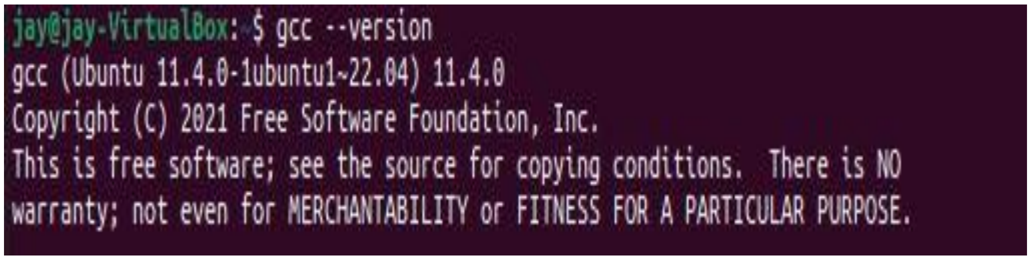
The above command will install the required packages for the GCC utility. Now, we can use the GCC utility in our machine.

Consider the snap of output on next page:



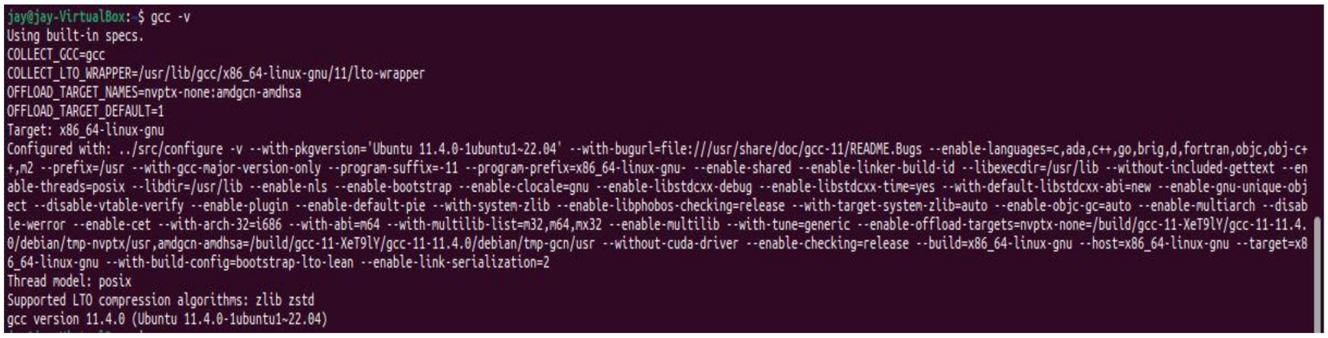
```
jay@jay-VirtualBox:~$ sudo apt install build-essential
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  binutils binutils-common binutils-x86-64-linux-gnu dpkg-dev fakeroot g++ g++-11 gcc gcc-11 libalgorithm-diff-perl libalgorithm-diff-xs-perl libalgorithm-merge-perl libasan6 libbinutils libc-dev-bin
 libc-devtools libc6 libc6-dbg
  libc6-dev libcc1-0 libcrypt-dev libctf-nobfd0 libctf0 libdpkg-perl libfakeroot libfile-fcntllock-perl libgcc-11-dev libitm1 liblsan0 libnsl-dev libquadmath0 libstdc++-11-dev libtirpc-dev libtsan0 l
ibubsan1 linux-libc-dev
  lto-disabled-list make manpages-dev rpcsvc-proto
Suggested packages:
  binutils-doc debian-keyring g++-multilib g++-11-multilib gcc-11-doc gcc-multilib autoconf automake libtool flex bison gcc-doc gcc-11-multilib gcc-11-locales glibc-doc git bzr libstdc++-11-doc make-
doc
Recommended packages:
  libnss-nis libnss-nisplus
The following NEW packages will be installed:
  binutils binutils-common binutils-x86-64-linux-gnu build-essential dpkg-dev fakeroot g++ g++-11 gcc gcc-11 libalgorithm-diff-perl libalgorithm-diff-xs-perl libalgorithm-merge-perl libasan6 libbinut
ils libc-dev-bin libc-devtools
  libc6-dev libcc1-0 libcrypt-dev libctf-nobfd0 libctf0 libdpkg-perl libfakeroot libfile-fcntllock-perl libgcc-11-dev libitm1 liblsan0 libnsl-dev libquadmath0 libstdc++-11-dev libtirpc-dev libtsan0 l
ibubsan1 linux-libc-dev
  lto-disabled-list make manpages-dev rpcsvc-proto
The following packages will be upgraded:
  libc6 libc6-dbg
2 upgraded, 39 newly installed, 0 to remove and 201 not upgraded.
Need to get 71.3 MB of archives.
After this operation, 186 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libc6-dbg amd64 2.35-0ubuntu3.6 [13.8 MB]
11% [1 libc6-dbg 9,865 kB/13.8 MB 71%]
Get:2 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libc6 amd64 2.35-0ubuntu3.6 [3,236 kB]
Get:3 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 binutils-common amd64 2.38-4ubuntu2.5 [222 kB]
Get:4 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libbinutils amd64 2.38-4ubuntu2.5 [662 kB]
Get:5 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libctf-nobfd0 amd64 2.38-4ubuntu2.5 [108 kB]
Get:6 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libctf0 amd64 2.38-4ubuntu2.5 [103 kB]
Get:7 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 binutils-x86-64-linux-gnu amd64 2.38-4ubuntu2.5 [2,326 kB]
Get:8 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 binutils amd64 2.38-4ubuntu2.5 [3,202 B]
Get:9 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libc-dev-bin amd64 2.35-0ubuntu3.6 [20.3 kB]
Get:10 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 linux-libc-dev amd64 5.15.0-94.104 [1,355 kB]
Get:11 http://in.archive.ubuntu.com/ubuntu jammy/main amd64 libcrypt-dev amd64 1:4.4.27-1 [112 kB]
Get:12 http://in.archive.ubuntu.com/ubuntu jammy/main amd64 rpcsvc-proto amd64 1.4.2-0ubuntu6 [68.5 kB]
Get:13 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libtirpc-dev amd64 1.3.2-2ubuntu0.1 [192 kB]
Get:14 http://in.archive.ubuntu.com/ubuntu jammy/main amd64 libnsl-dev amd64 1.3.0-2build2 [71.3 kB]
Get:15 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libc6-dev amd64 2.35-0ubuntu3.6 [2,100 kB]
Get:16 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libcc1-0 amd64 12.3.0-1ubuntu1~22.04 [48.3 kB]
Get:17 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libitm1 amd64 12.3.0-1ubuntu1~22.04 [30.2 kB]
Get:18 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libasan6 amd64 11.4.0-1ubuntu1~22.04 [2,282 kB]
Get:19 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 liblsan0 amd64 12.3.0-1ubuntu1~22.04 [1,069 kB]
Get:20 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libtsan0 amd64 11.4.0-1ubuntu1~22.04 [2,260 kB]
Get:21 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libubsan1 amd64 12.3.0-1ubuntu1~22.04 [976 kB]
Get:22 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libquadmath0 amd64 12.3.0-1ubuntu1~22.04 [154 kB]
Get:23 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libgcc-11-dev amd64 11.4.0-1ubuntu1~22.04 [2,517 kB]
```

Step 3: Verify the installation.
To verify the installation, execute the gcc -version command as follows:
$gcc --version



```
jay@jay-VirtualBox:~$ gcc --version
gcc (Ubuntu 11.4.0-1ubuntu1~22.04) 11.4.0
Copyright (C) 2021 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.
```

It will display the installed version of GCC utility. To display the more specific details about the version, use the '-v' option.
Consider the output on next page:

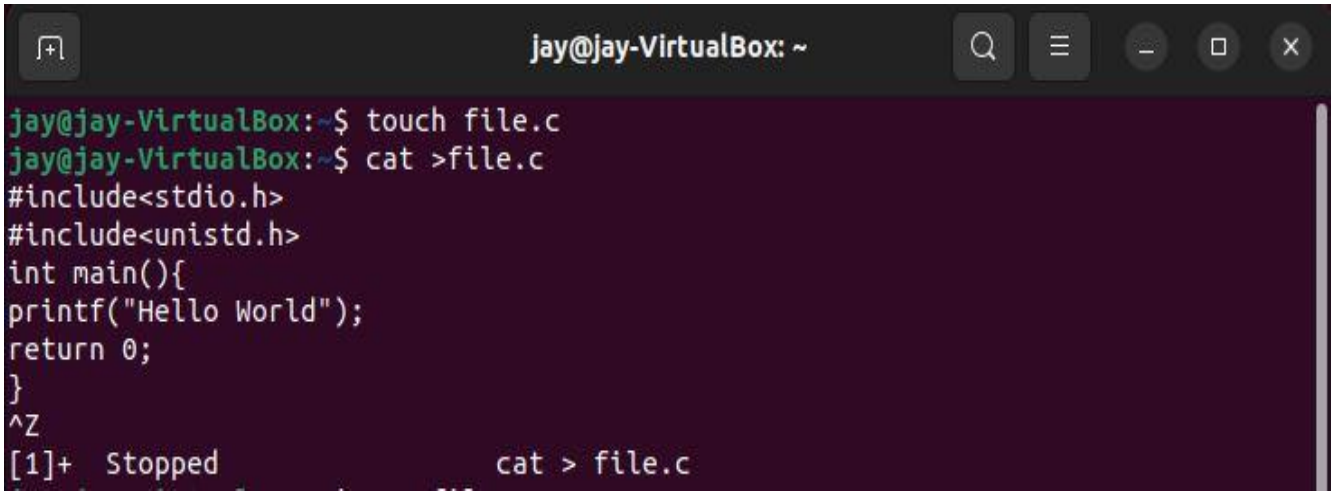


```
jay@jay-VirtualBox:~$ gcc -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/lib/gcc/x86_64-linux-gnu/11/lto-wrapper
OFFLOAD_TARGET_NAMES=nvptx-none:amdgcn-amdhsa
OFFLOAD_TARGET_DEFAULT=1
Target: x86_64-linux-gnu
Configured with: ../src/configure -v --with-pkgversion='Ubuntu 11.4.0-1ubuntu1~22.04' --with-bugurl=file:///usr/share/doc/gcc-11/README.Bugs --enable-languages=c,ada,c++,go,brig,d,fortran,objc,obj-c+
+,m2 --prefix=/usr --with-gcc-major-version-only --program-suffix=-11 --program-prefix=x86_64-linux-gnu- --enable-shared --enable-linker-build-id --libexecdir=/usr/lib --without-included-gettext --en
able-threads=posix --libdir=/usr/lib --enable-nls --enable-bootstrap --enable-clocale=gnu --enable-libstdcxx-debug --enable-libstdcxx-time=yes --with-default-libstdcxx-abi=new --enable-gnu-unique-obj
ect --disable-vtable-verify --enable-plugin --enable-default-pie --with-system-zlib --enable-libphobos-checking=release --with-target-system-zlib=auto --enable-objc-gc=auto --enable-multiarch --disab
le-werror --enable-cet --with-arch-32=i686 --with-abi=m64 --with-multilib-list=m32,m64,mx32 --enable-multilib --with-tune=generic --enable-offload-targets=nvptx-none=/build/gcc-11-XeT9lY/gcc-11-11.4.
0/debian/tmp-nvptx/usr,amdgcn-amdhsa=/build/gcc-11-XeT9lY/gcc-11-11.4.0/debian/tmp-gcn/usr --without-cuda-driver --enable-checking=release --build=x86_64-linux-gnu --host=x86_64-linux-gnu --target=x8
6_64-linux-gnu --with-build-config=bootstrap-lto-lean --enable-link-serialization=2
Thread model: posix
Supported LTO compression algorithms: zlib zstd
gcc version 11.4.0 (Ubuntu 11.4.0-1ubuntu1~22.04)
```

Here, we have successfully installed the GCC utility. Let's understand how to use it. We will create and execute some c programs using GCC. Run the first C program by gcc Create a basic c program "Hello World".

Create a file 'file.c" and put the below code in it:

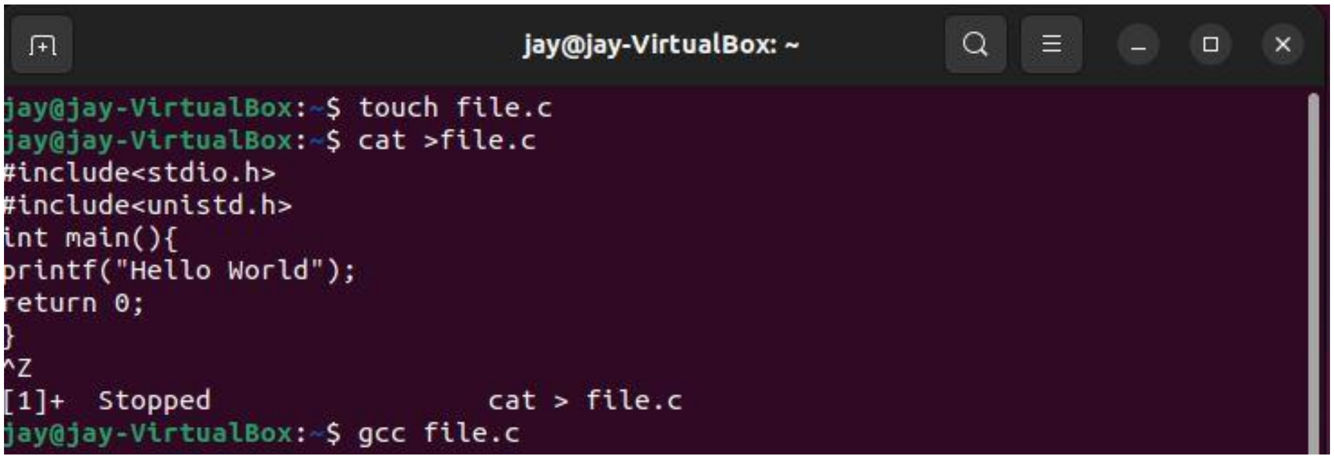
```
 #include<stdio.h>
#include<unistd.h>
 int main() {
 printf("Hello World");
 return 0;
 }
```



```
jay@jay-VirtualBox:~$ touch file.c
jay@jay-VirtualBox:~$ cat >file.c
#include<stdio.h>
#include<unistd.h>
int main(){
printf("Hello World");
return 0;
}
^Z
[1]+  Stopped                 cat > file.c
```

Now, compile the file.c

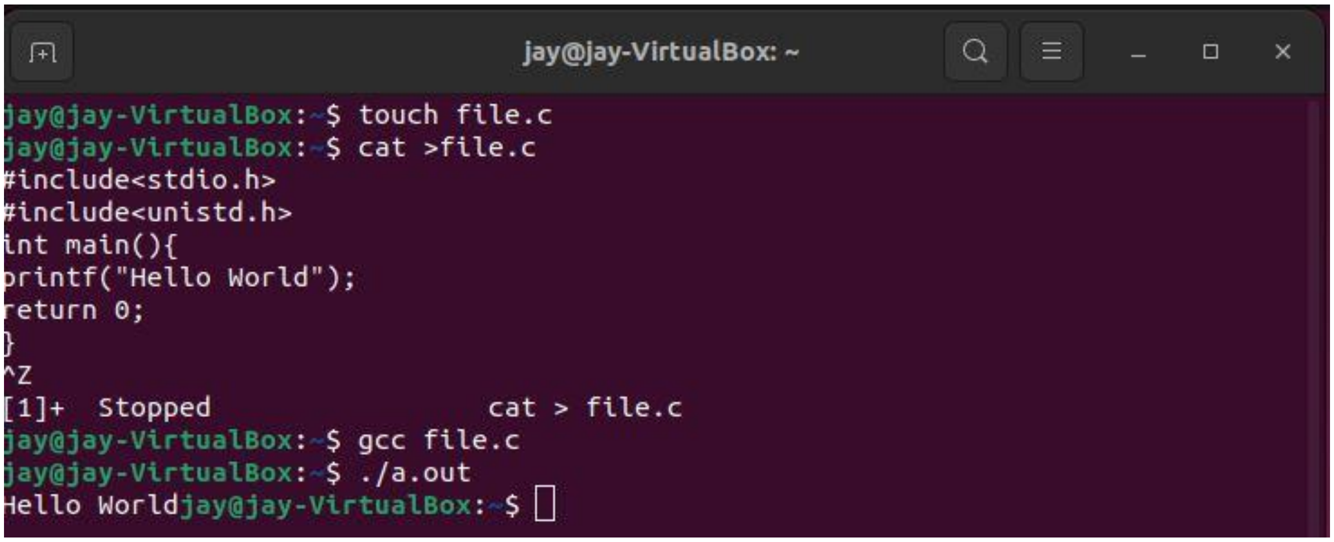
1) gcc file.c



```
jay@jay-VirtualBox:~$ touch file.c
jay@jay-VirtualBox:~$ cat >file.c
#include<stdio.h>
#include<unistd.h>
int main(){
printf("Hello World");
return 0;
}
^Z
[1]+  Stopped                 cat > file.c
jay@jay-VirtualBox:~$ gcc file.c
```

2) ./a.out

Consider the below output



```
jay@jay-VirtualBox:~$ touch file.c
jay@jay-VirtualBox:~$ cat >file.c
#include<stdio.h>
#include<unistd.h>
int main(){
printf("Hello World");
return 0;
}
^Z
[1]+  Stopped                 cat > file.c
jay@jay-VirtualBox:~$ gcc file.c
jay@jay-VirtualBox:~$ ./a.out
Hello Worldjay@jay-VirtualBox:~$ 
```

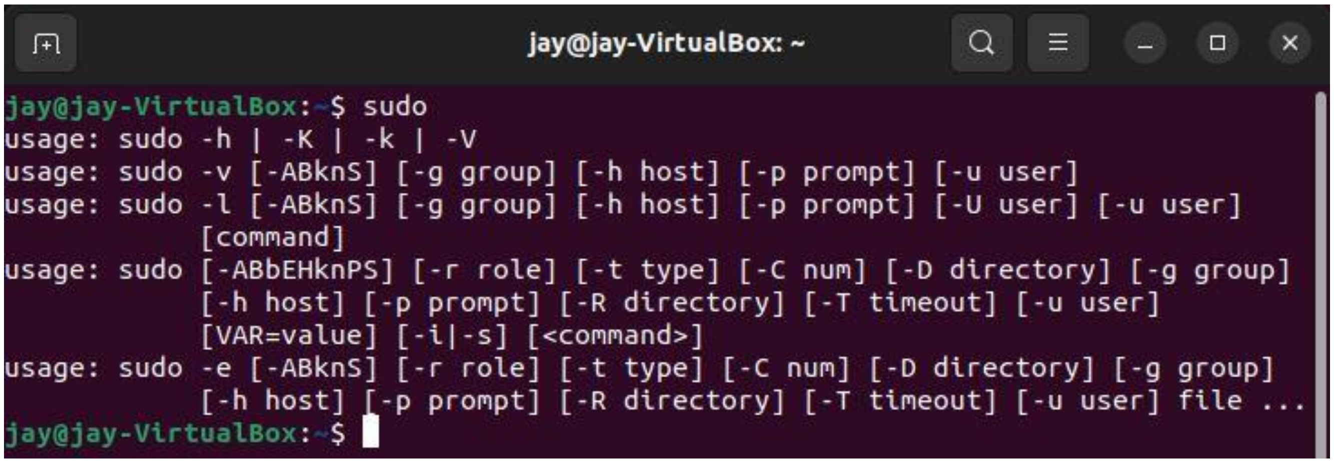
Program 2: Implement the basic and user status commands like: su, sudo, man, help, history, who, whoami, id, uname, uptime, free, tty, cal, date, hostname, reboot, clear.

Introduction:
Linux is famous for its powerful commands. To use Linux effectively, all users should know how to use terminal commands. Although the OS has a GUI, many functionalities work faster when run as commands through the terminal.

Prerequisites:
1. A system running Linux.
2. Access to the command line/terminal.

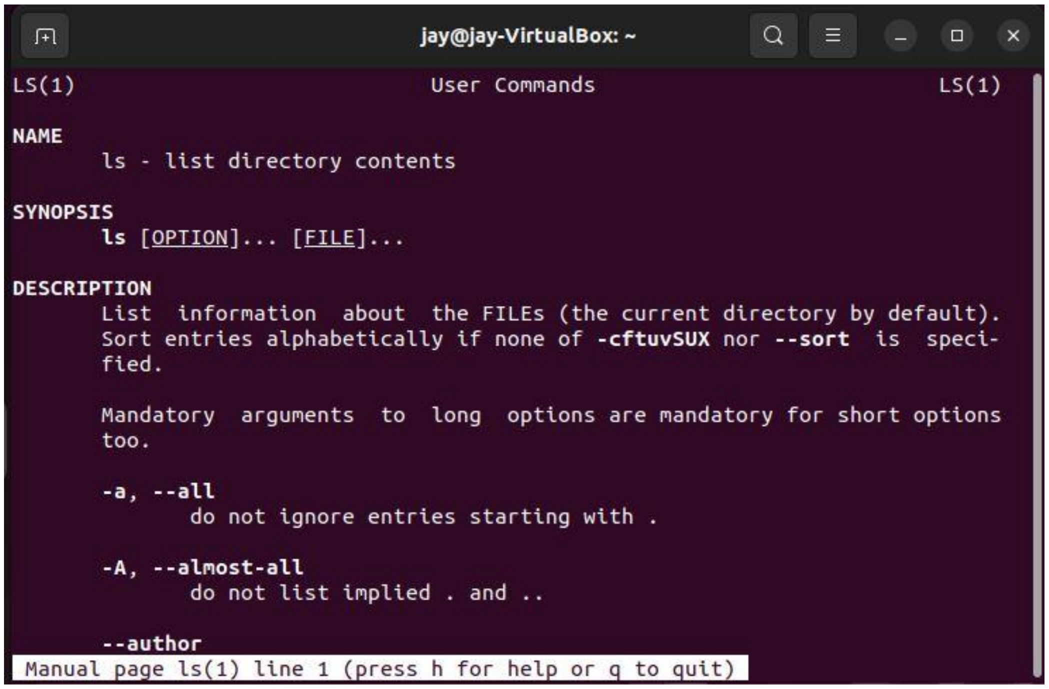
Basic Linux Commands:

1. **sudo command-** It allows you to run programs with the security privileges of another user(by default, as the superuser).
   Syntax: sudo



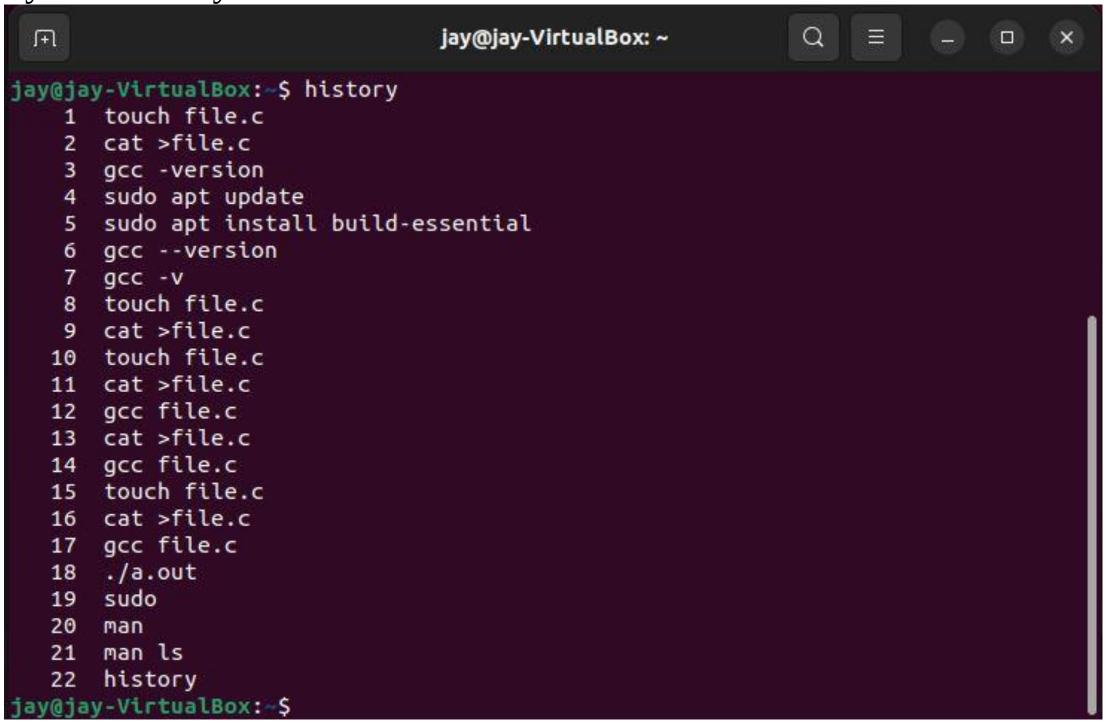
```
jay@jay-VirtualBox:~$ sudo
usage: sudo -h | -K | -k | -V
usage: sudo -v [-ABknS] [-g group] [-h host] [-p prompt] [-u user]
usage: sudo -l [-ABknS] [-g group] [-h host] [-p prompt] [-U user] [-u user]
            [command]
usage: sudo [-ABbEHknPS] [-r role] [-t type] [-C num] [-D directory] [-g group]
            [-h host] [-p prompt] [-R directory] [-T timeout] [-u user]
            [VAR=value] [-i|-s] [<command>]
usage: sudo -e [-ABknS] [-r role] [-t type] [-C num] [-D directory] [-g group]
            [-h host] [-p prompt] [-R directory] [-T timeout] [-u user] file ...
jay@jay-VirtualBox:~$ 
```

2. **man command-** It is used to display the user manual of any command that we can run on the terminal.
   Syntax: man [command]
   Example: man ls



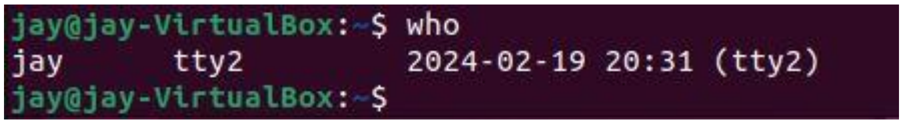
```
LS(1)                            User Commands                           LS(1)

NAME
       ls - list directory contents

SYNOPSIS
       ls [OPTION]... [FILE]...

DESCRIPTION
       List  information  about  the FILEs (the current directory by default).
       Sort entries alphabetically if none of -cftuvSUX nor --sort  is  speci-
       fied.

       Mandatory  arguments  to  long  options are mandatory for short options
       too.

       -a, --all
              do not ignore entries starting with .

       -A, --almost-all
              do not list implied . and ..

       --author
 Manual page ls(1) line 1 (press h for help or q to quit)
```

3. **history command-** This feature allows users to recall, reuse, and modify commands without having to retype them.
   Syntax: history



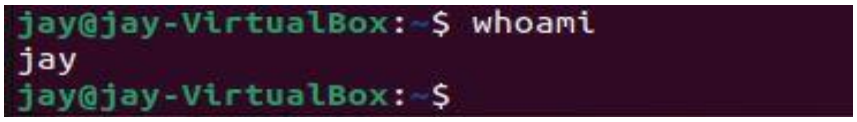
```
jay@jay-VirtualBox:~$ history
    1  touch file.c
    2  cat >file.c
    3  gcc -version
    4  sudo apt update
    5  sudo apt install build-essential
    6  gcc --version
    7  gcc -v
    8  touch file.c
    9  cat >file.c
   10  touch file.c
   11  cat >file.c
   12  gcc file.c
   13  cat >file.c
   14  gcc file.c
   15  touch file.c
   16  cat >file.c
   17  gcc file.c
   18  ./a.out
   19  sudo
   20  man
   21  man ls
   22  history
jay@jay-VirtualBox:~$
```

4. **who command-** It is a tool that prints information about users who are currently logged in. who command only sees a real user who logged in.
   Syntax: who



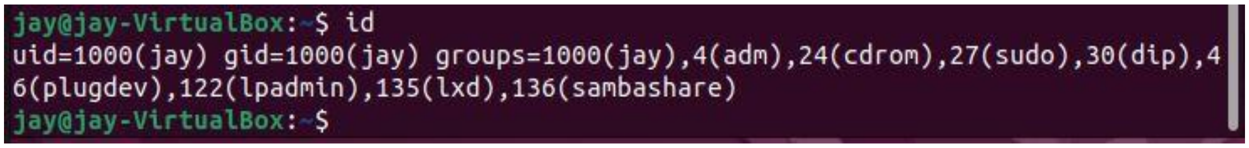
```
jay@jay-VirtualBox:~$ who
jay      tty2         2024-02-19 20:31 (tty2)
jay@jay-VirtualBox:~$
```

5. **whoami command-** The command whoami in Linux is the concatenation of the words "Who am I**?**" It displays the username of the effective user associated with the current shell session.
   Syntax: whoami



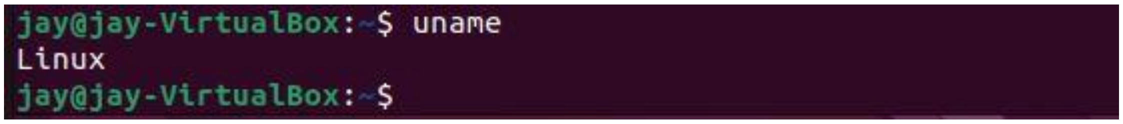
```
jay@jay-VirtualBox:~$ whoami
jay
jay@jay-VirtualBox:~$
```

6. **id command-** The id command in Linux is used for displaying the real and effective user ID and group ID of a user.
   Syntax: id



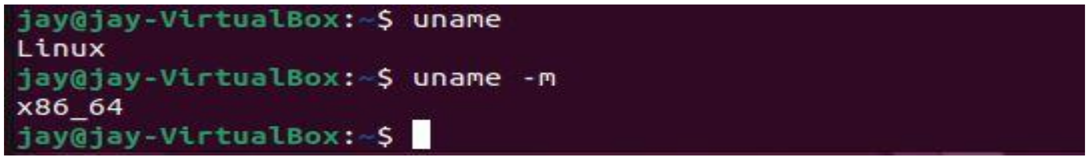
```
jay@jay-VirtualBox:~$ id
uid=1000(jay) gid=1000(jay) groups=1000(jay),4(adm),24(cdrom),27(sudo),30(dip),4
6(plugdev),122(lpadmin),135(lxd),136(sambashare)
jay@jay-VirtualBox:~$
```

7. **uname command-** The uname command writes to standard output the name of the operating system that you are using.
   Syntax: uname



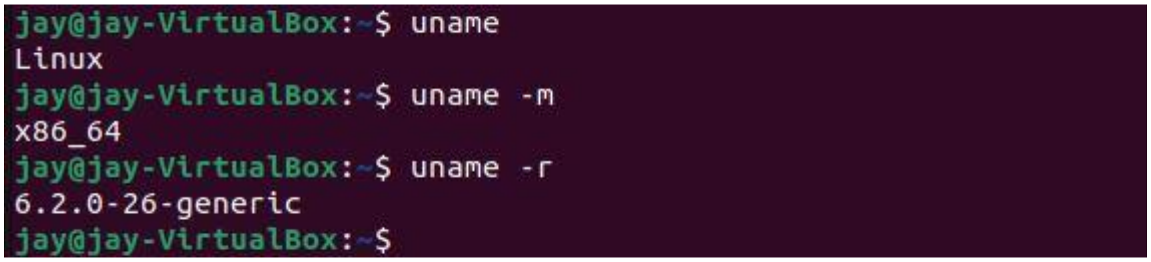
```
jay@jay-VirtualBox:~$ uname
Linux
jay@jay-VirtualBox:~$
```

   There are additional options which provide flexibility with the display output.
   - **uname -m-** Displays the machine ID number of the hardware running the system.



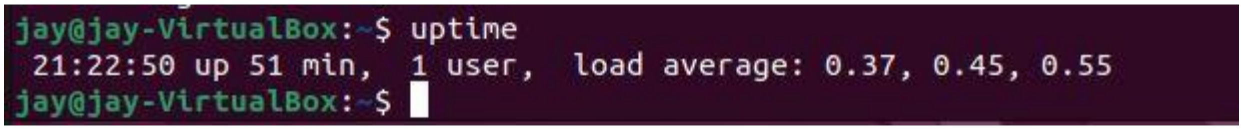
```
jay@jay-VirtualBox:~$ uname
Linux
jay@jay-VirtualBox:~$ uname -m
x86_64
jay@jay-VirtualBox:~$
```

- **uname -r-** Displays the release number of the operating system.



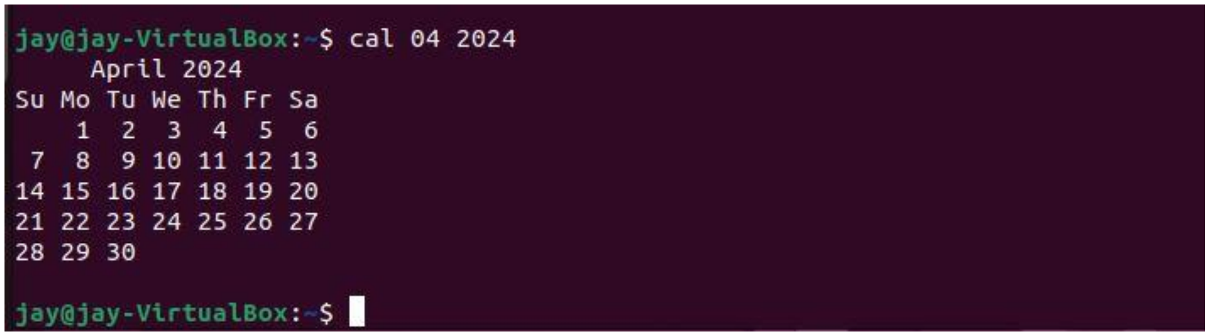
```
jay@jay-VirtualBox:~$ uname
Linux
jay@jay-VirtualBox:~$ uname -m
x86_64
jay@jay-VirtualBox:~$ uname -r
6.2.0-26-generic
jay@jay-VirtualBox:~$
```

8. **uptime command-** It is used to find out how long the system is active (running).
   Syntax: uptime



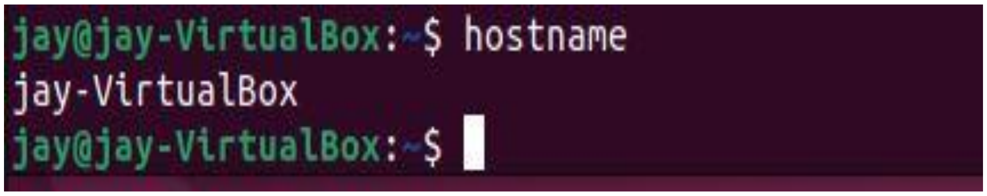
```
jay@jay-VirtualBox:~$ uptime
 21:22:50 up 51 min,  1 user,  load average: 0.37, 0.45, 0.55
jay@jay-VirtualBox:~$
```

9. **cal command-** It is a calendar command in Linux which is used to see the calendar of a specific month or a whole year.
   Syntax: cal [[month] year]



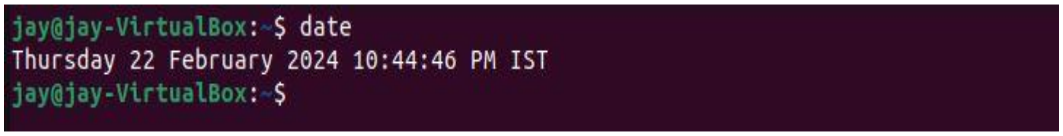
```
jay@jay-VirtualBox:~$ cal 04 2024
     April 2024
Su Mo Tu We Th Fr Sa
    1  2  3  4  5  6
 7  8  9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30

jay@jay-VirtualBox:~$
```

10. **hostname command-** The Linux hostname command allows us to set and view the hostname of the system. A hostname is the name of any computer that is connected to a network that is uniquely identified over a network. It can be accessed without using a particular IP address.
    Syntax: hostname



```
jay@jay-VirtualBox:~$ hostname
jay-VirtualBox
jay@jay-VirtualBox:~$
```

11. **date command-** It is used to display the system date and time.
    Syntax: date



```
jay@jay-VirtualBox:~$ date
Thursday 22 February 2024 10:44:46 PM IST
jay@jay-VirtualBox:~$
```

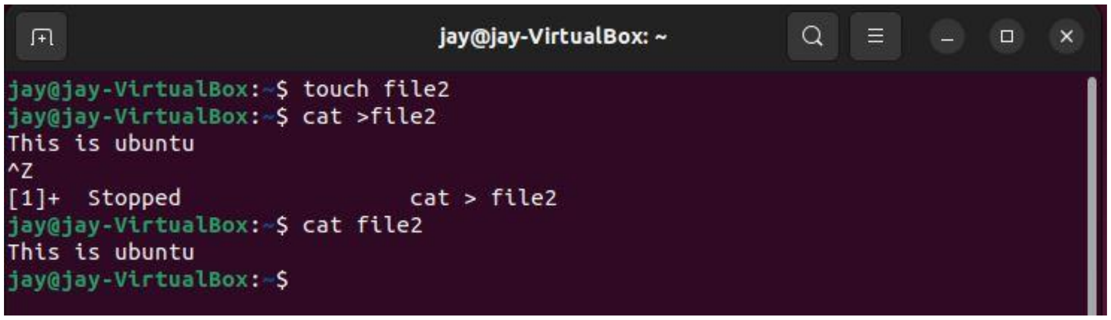
12. **clear command-** It is a standard Unix computer operating system command that is used to clear the terminal screen.
    Syntax: clear

13. **reboot command-** It is used to restart or reboot the system. In a Linux system administration, there comes a need to restart the server after the completion of some network and other major updates. It can be software or hardware that is being carried on the server. Rebooting is needed so that the changes that the user has made can affect the server.
    Syntax: reboot

Program 3: Implement the commands that is used for Creating and Manipulating files: cat, cp, mv, rm, ls and its options, touch and their options, which is, where is, what is,the basics of the Unix file system.

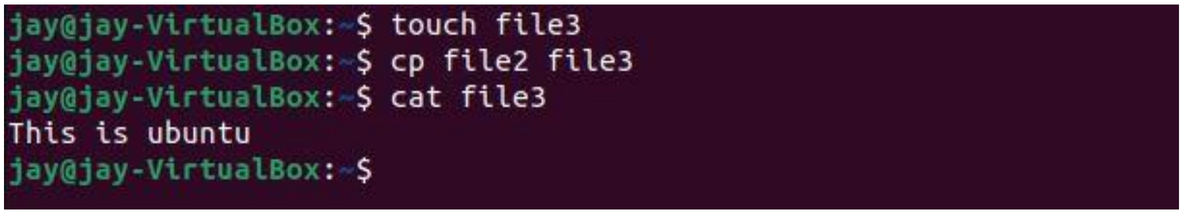
Commands:

1. **cat command-** The cat command on Linux concatenates files together. It's often used to concatenate one file to nothing to print the single file's contents to the terminal. This is a quick way to preview the contents of a text file without having to open the file in a large application.
   Syntax: cat file1



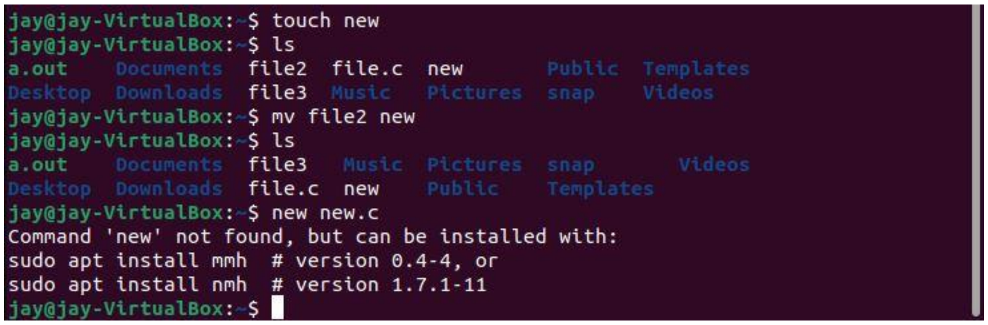
```
jay@jay-VirtualBox:~$ touch file2
jay@jay-VirtualBox:~$ cat >file2
This is ubuntu
^Z
[1]+  Stopped                 cat > file2
jay@jay-VirtualBox:~$ cat file2
This is ubuntu
jay@jay-VirtualBox:~$
```

2. **cp command-** The cp command copies the source file specified by the SourceFile parameter to the destination file specified by the TargetFile parameter. If the target file exists, cp overwrites the contents, but the mode, owner, and group associated with it are not changed.
   Syntax: cp [file1][file2]



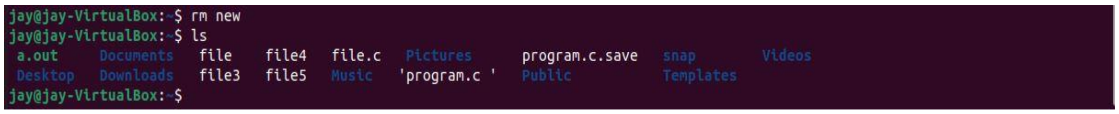
```
jay@jay-VirtualBox:~$ touch file3
jay@jay-VirtualBox:~$ cp file2 file3
jay@jay-VirtualBox:~$ cat file3
This is ubuntu
jay@jay-VirtualBox:~$
```

3. **mv command-** The mv command moves files and directories from one directory to another or renames a file or directory. If you move a file or directory to a new directory, it retains the base file name. When you move a file, all links to other files remain intact, except when you move it to a different file system.
   Syntax: mv [file1][file2]



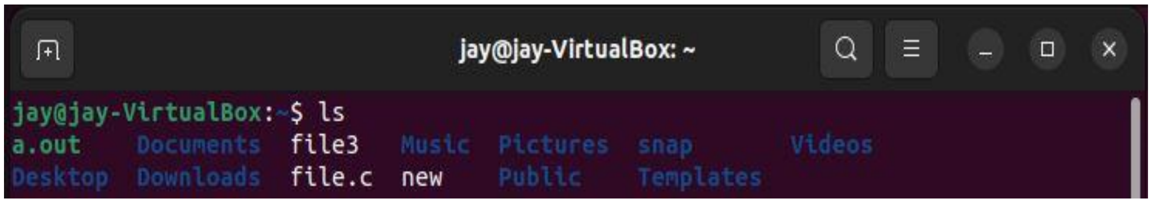
```
jay@jay-VirtualBox:~$ touch new
jay@jay-VirtualBox:~$ ls
a.out     Documents  file2  file.c  new       Public  Templates
Desktop   Downloads  file3  Music   Pictures  snap    Videos
jay@jay-VirtualBox:~$ mv file2 new
jay@jay-VirtualBox:~$ ls
a.out     Documents  file3   Music  Pictures  snap       Videos
Desktop   Downloads  file.c  new    Public    Templates
jay@jay-VirtualBox:~$ new new.c
Command 'new' not found, but can be installed with:
sudo apt install mmh  # version 0.4-4, or
sudo apt install nmh  # version 1.7.1-11
jay@jay-VirtualBox:~$ 
```

4. **rm command-** It is a general command in Unix and other Unix-like systems. It deletes objects like symbolic links, directories, and computer files from the file systems.
   Syntax: rm [file]



```
jay@jay-VirtualBox:~$ rm new
jay@jay-VirtualBox:~$ ls
 a.out     Documents  file   file4  file.c  Pictures      program.c.save  snap       Videos
 Desktop   Downloads  file3  file5  Music  'program.c '   Public          Templates
jay@jay-VirtualBox:~$
```

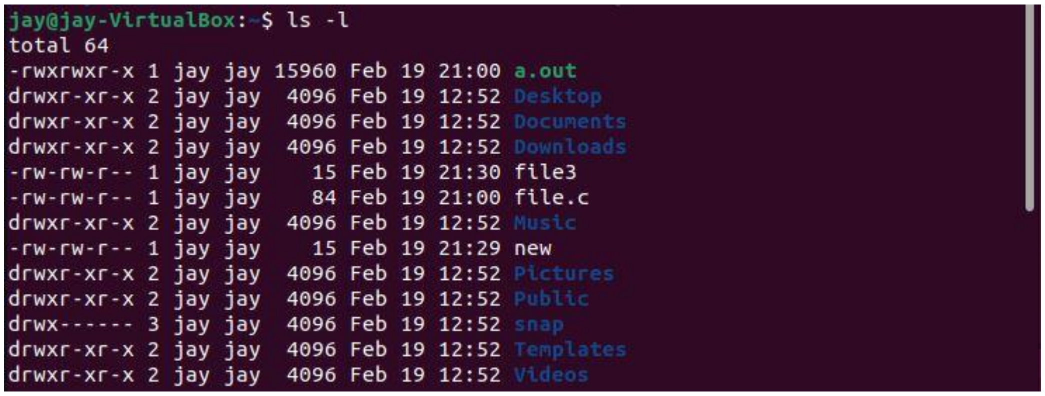
5. **ls command-** The ls command is one of the more basic commands in Linux. It is designed to list the names and features of files and directories. It can be used for a single file or as many as all files and folders in a selected set of directories.
   Syntax: ls



```
jay@jay-VirtualBox:~$ ls
a.out     Documents  file3   Music  Pictures  snap       Videos
Desktop   Downloads  file.c  new    Public    Templates
```

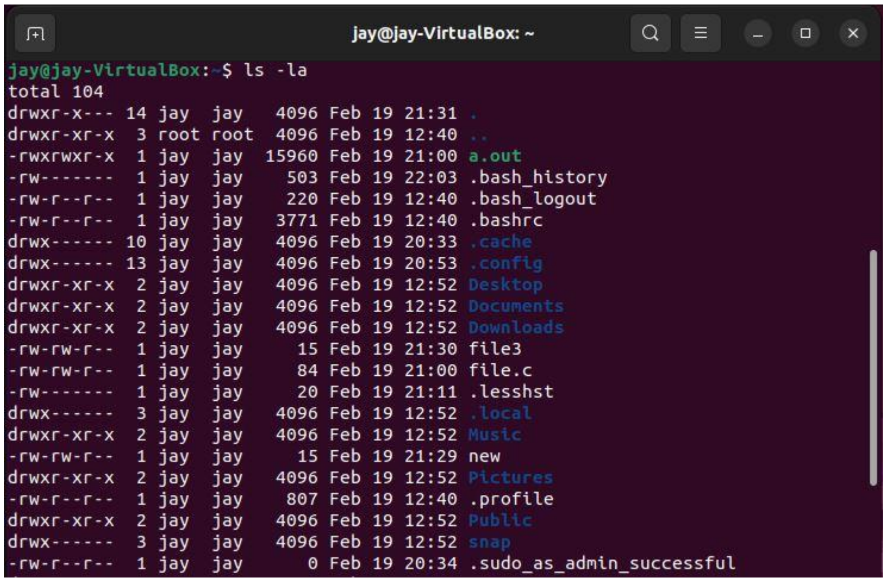
There are additional options which provide flexibility with the display output:

- ls -l: It gives the output as a list.



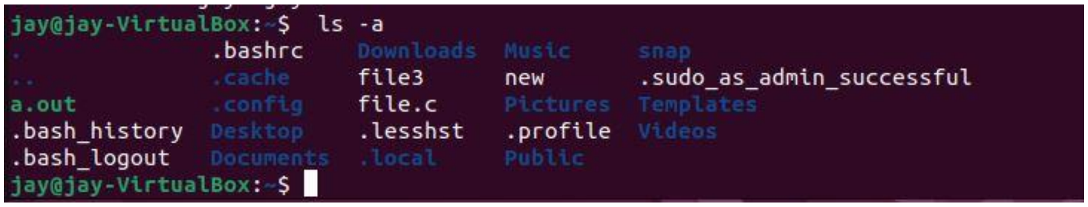
```
jay@jay-VirtualBox:~$ ls -l
total 64
-rwxrwxr-x 1 jay jay 15960 Feb 19 21:00 a.out
drwxr-xr-x 2 jay jay  4096 Feb 19 12:52 Desktop
drwxr-xr-x 2 jay jay  4096 Feb 19 12:52 Documents
drwxr-xr-x 2 jay jay  4096 Feb 19 12:52 Downloads
-rw-rw-r-- 1 jay jay    15 Feb 19 21:30 file3
-rw-rw-r-- 1 jay jay    84 Feb 19 21:00 file.c
drwxr-xr-x 2 jay jay  4096 Feb 19 12:52 Music
-rw-rw-r-- 1 jay jay    15 Feb 19 21:29 new
drwxr-xr-x 2 jay jay  4096 Feb 19 12:52 Pictures
drwxr-xr-x 2 jay jay  4096 Feb 19 12:52 Public
drwx------ 3 jay jay  4096 Feb 19 12:52 snap
drwxr-xr-x 2 jay jay  4096 Feb 19 12:52 Templates
drwxr-xr-x 2 jay jay  4096 Feb 19 12:52 Videos
```

- ls -la: It gives the output as a list but also include some hidden files.



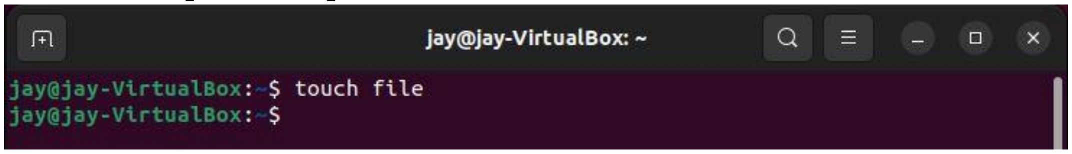
```
jay@jay-VirtualBox:~$ ls -la
total 104
drwxr-x--- 14 jay  jay   4096 Feb 19 21:31 .
drwxr-xr-x  3 root root  4096 Feb 19 12:40 ..
-rwxrwxr-x  1 jay  jay  15960 Feb 19 21:00 a.out
-rw-------  1 jay  jay    503 Feb 19 22:03 .bash_history
-rw-r--r--  1 jay  jay    220 Feb 19 12:40 .bash_logout
-rw-r--r--  1 jay  jay   3771 Feb 19 12:40 .bashrc
drwx------ 10 jay  jay   4096 Feb 19 20:33 .cache
drwx------ 13 jay  jay   4096 Feb 19 20:53 .config
drwxr-xr-x  2 jay  jay   4096 Feb 19 12:52 Desktop
drwxr-xr-x  2 jay  jay   4096 Feb 19 12:52 Documents
drwxr-xr-x  2 jay  jay   4096 Feb 19 12:52 Downloads
-rw-rw-r--  1 jay  jay     15 Feb 19 21:30 file3
-rw-rw-r--  1 jay  jay     84 Feb 19 21:00 file.c
-rw-------  1 jay  jay     20 Feb 19 21:11 .lesshst
drwx------  3 jay  jay   4096 Feb 19 12:52 .local
drwxr-xr-x  2 jay  jay   4096 Feb 19 12:52 Music
-rw-rw-r--  1 jay  jay     15 Feb 19 21:29 new
drwxr-xr-x  2 jay  jay   4096 Feb 19 12:52 Pictures
-rw-r--r--  1 jay  jay    807 Feb 19 12:40 .profile
drwxr-xr-x  2 jay  jay   4096 Feb 19 12:52 Public
drwx------  3 jay  jay   4096 Feb 19 12:52 snap
-rw-r--r--  1 jay  jay      0 Feb 19 20:34 .sudo_as_admin_successful
```

- ls -a: In Linux, hidden files start with. (dot) symbol and they are not visible in the regular directory. The (ls -a) command will enlist the whole list of the current directory including the hidden files.



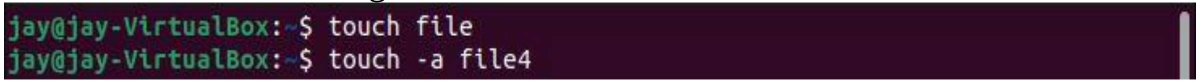
```
jay@jay-VirtualBox:~$  ls -a
.               .bashrc     Downloads  Music     snap
..              .cache      file3      new       .sudo_as_admin_successful
a.out           .config     file.c     Pictures  Templates
.bash_history   Desktop     .lesshst   .profile  Videos
.bash_logout    Documents   .local     Public
jay@jay-VirtualBox:~$
```

6. **touch command-** The touch command in Linux is used to create a new empty file and to change the timestamps of existing files. You can use it with the syntax: touch file. txt.
   Syntax: touch [filename]



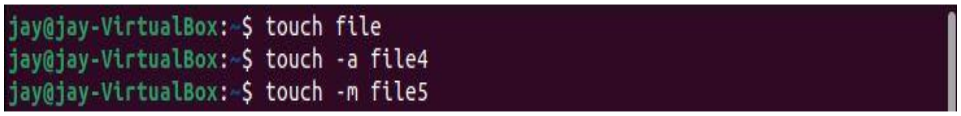
```
jay@jay-VirtualBox:~$ touch file
jay@jay-VirtualBox:~$
```

- touch -a: To change file access and modification time.



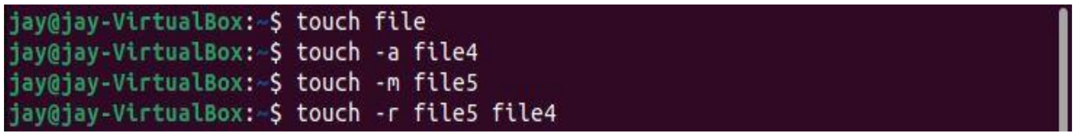
```
jay@jay-VirtualBox:~$ touch file
jay@jay-VirtualBox:~$ touch -a file4
```

- touch -m: It is used to only modify time of a file.



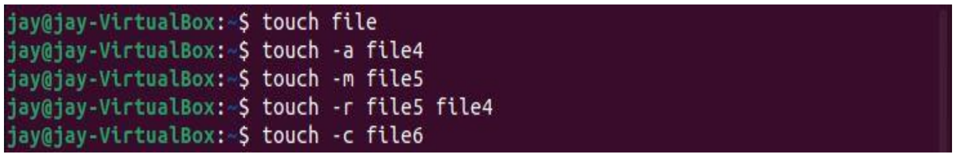
```
jay@jay-VirtualBox:~$ touch file
jay@jay-VirtualBox:~$ touch -a file4
jay@jay-VirtualBox:~$ touch -m file5
```

- touch -r: To update time of one file with reference to the other file.



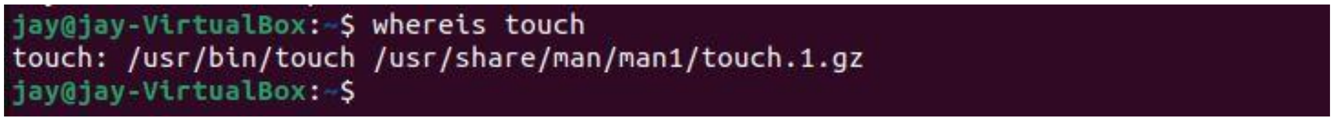
```
jay@jay-VirtualBox:~$ touch file
jay@jay-VirtualBox:~$ touch -a file4
jay@jay-VirtualBox:~$ touch -m file5
jay@jay-VirtualBox:~$ touch -r file5 file4
```

- touch -c: It doesn't create an empty file.



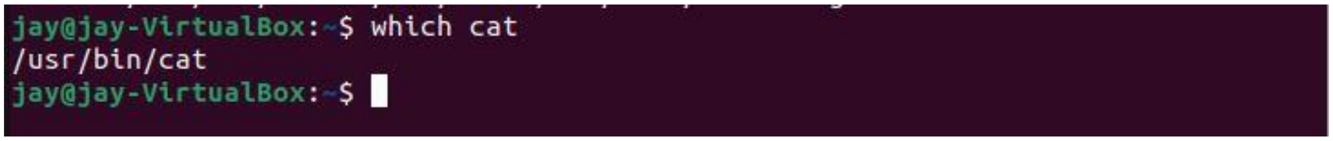
```
jay@jay-VirtualBox:~$ touch file
jay@jay-VirtualBox:~$ touch -a file4
jay@jay-VirtualBox:~$ touch -m file5
jay@jay-VirtualBox:~$ touch -r file5 file4
jay@jay-VirtualBox:~$ touch -c file6
```

7. **whereis command-** This command is used to find the location of source/binary file of a command and manuals sections for a specified file in Linux system.
   Syntax: whereis [command_name]



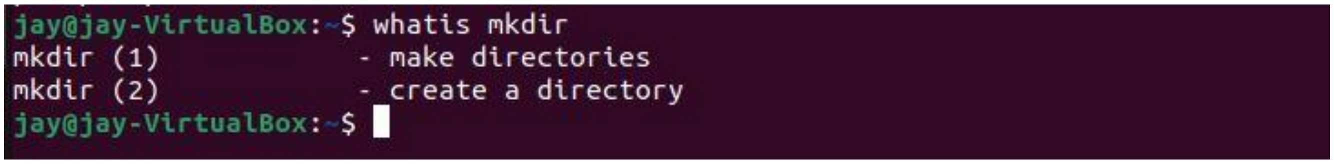
```
jay@jay-VirtualBox:~$ whereis touch
touch: /usr/bin/touch /usr/share/man/man1/touch.1.gz
jay@jay-VirtualBox:~$
```

8. **which command-** which command in Linux is a command that is used to locate the executable file associated with the given command by searching it in the path environment variable.
   Syntax: which [command_name]



```
jay@jay-VirtualBox:~$ which cat
/usr/bin/cat
jay@jay-VirtualBox:~$
```

9. **whatis command-** This command in Linux is used to get a one-line manual page description. In Linux, each manual page has some sort of description within it. So, this command search for the manual pages names and show the manual page description of the specified filename or argument.
   Syntax: whatis [command_name]

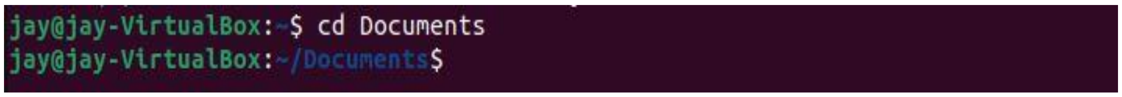


```
jay@jay-VirtualBox:~$ whatis mkdir
mkdir (1)            - make directories
mkdir (2)            - create a directory
jay@jay-VirtualBox:~$
```

Program 4: Implement Directory oriented commands: cd, pwd, mkdir, rmdir, Comparing Files using diff, cmp, comm.

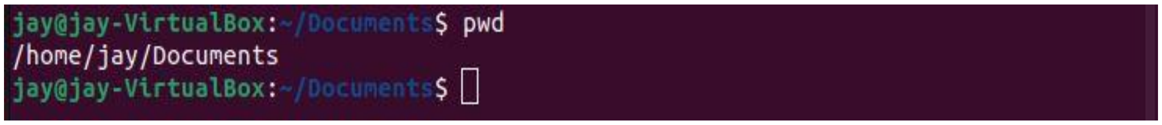
Directory- A directory is a file the sole job of which is to store the file names and the related information. All the files, whether ordinary, special, or directory, are contained in directories.

1. **cd command:** It is a shell built-in command for changing the current working directory.
   Syntax: cd<directory>



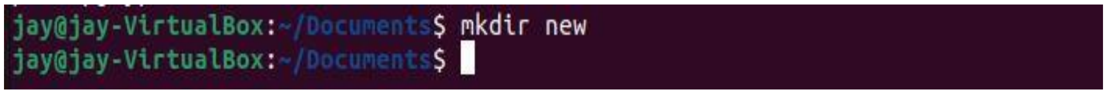
```
jay@jay-VirtualBox:~$ cd Documents
jay@jay-VirtualBox:~/Documents$
```

2. **pwd command-** It is a shell built-in command that prints the current location. The output shows an absolute directory path.
   Syntax: pwd



```
jay@jay-VirtualBox:~/Documents$ pwd
/home/jay/Documents
jay@jay-VirtualBox:~/Documents$
```

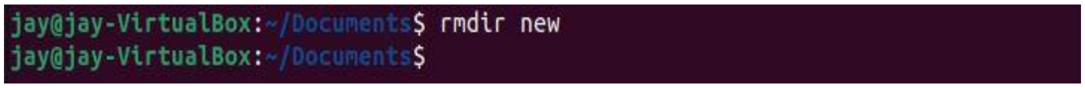
3. **mkdir command-** The mkdir command creates directories. This command can create multiple directories at once as well as set the permissions for the directories. It is important to note that the user executing this command must have enough permissions to create a directory in the parent directory.
   Syntax: mkdir [directory names]



```
jay@jay-VirtualBox:~/Documents$ mkdir new
jay@jay-VirtualBox:~/Documents$
```

   This command has created a directory named new.

4. **rmdir command-** The rmdir(remove directory) command is used to delete an empty directory. This command deletes the directory named new.
   Syntax: rmdir [directory name]



```
jay@jay-VirtualBox:~/Documents$ rmdir new
jay@jay-VirtualBox:~/Documents$
```

Program 5 : Write a program to create and execute process using fork( ) and exec( ) system calls.

What is fork( ) system call?

The fork system call is used to create a new processes. The newly created process is the child process. The process which calls fork and creates a new process is the parent process. The child and parent processes are executed concurrently.

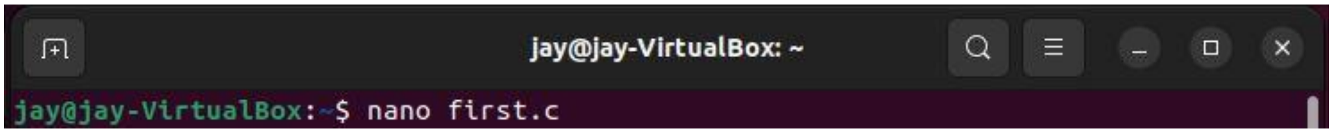
But the child and parent processes reside on different memory spaces. These memory spaces have same content and whatever operation is performed by one process will not affect the other process.

When the child processes is created; now both the processes will have the same Program Counter (PC), so both of these processes will point to the same next instruction. The files opened by the parent process will be the same for child process.

Creation and Execution using fork( ):

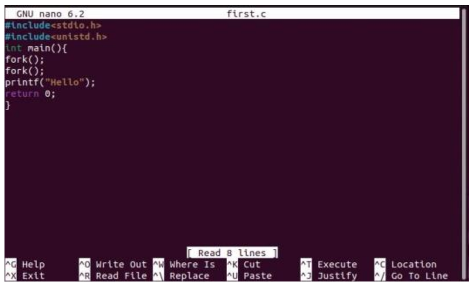
Step 1-Firstly write your program in c. For this we will use the nano text editor in the command-line user interface to write our program. Type the following command in the terminal.

nano <file_name_with_extension>



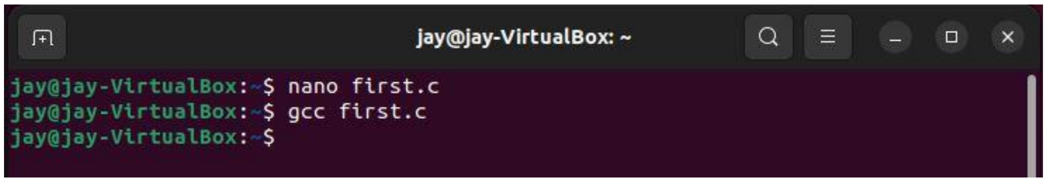
```
jay@jay-VirtualBox:~$ nano first.c
```

Step 2- Now press enter and write a c program.To exit the editor press Ctrl+X.



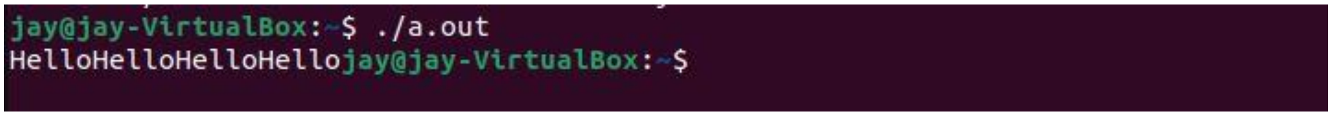
```
#include<stdio.h>
#include<unistd.h>
int main(){
fork();
fork();
printf("Hello");
return 0;
}
```

Step 3- Now compile the code using command “gcc <file_name_with_extension>



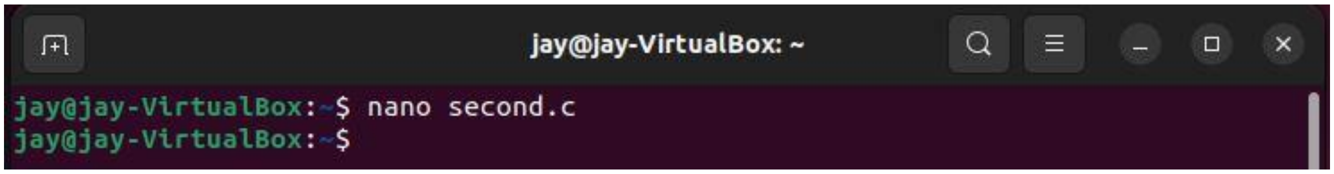
```
jay@jay-VirtualBox:~$ nano first.c
jay@jay-VirtualBox:~$ gcc first.c
jay@jay-VirtualBox:~$
```

Step 4- Now to get the run the code use command "./a.out".

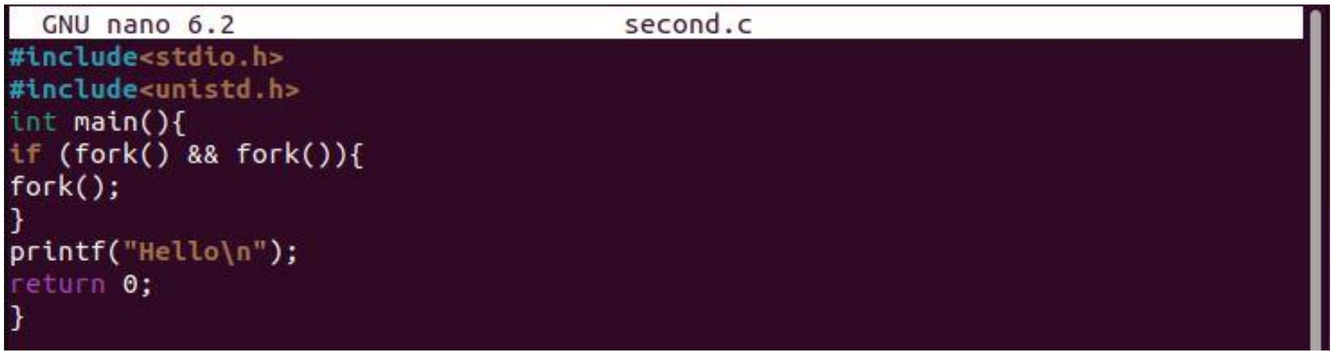


```
jay@jay-VirtualBox:~$ ./a.out
HelloHelloHelloHellojay@jay-VirtualBox:~$
```

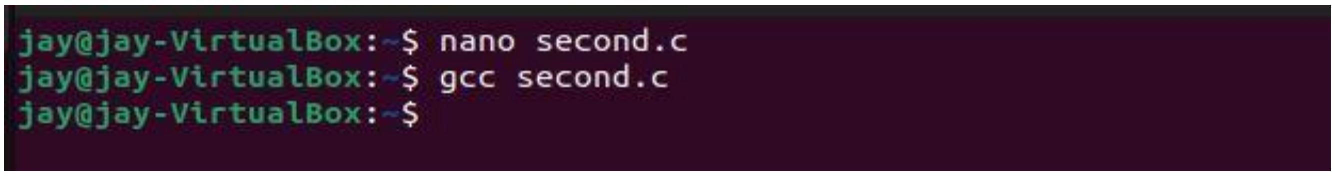
Let's take one more example of fork():



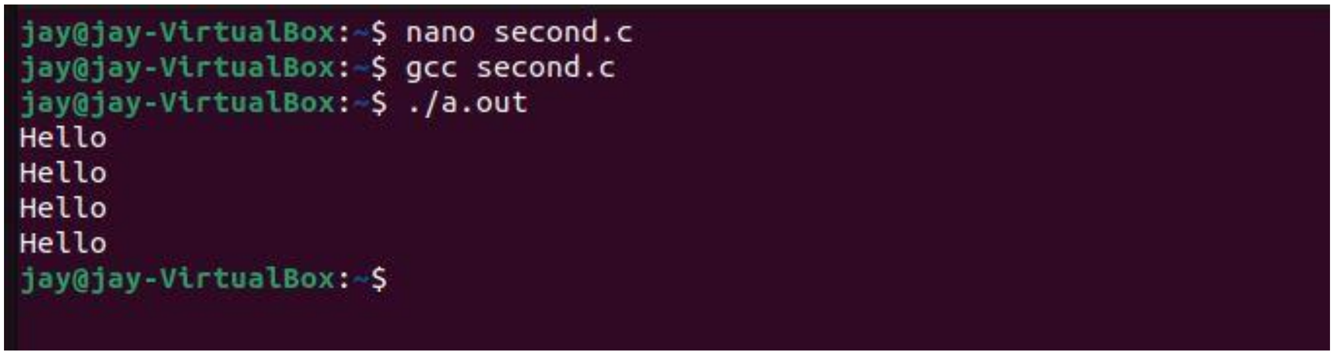
```
jay@jay-VirtualBox:~$ nano second.c
jay@jay-VirtualBox:~$
```



```
  GNU nano 6.2                    second.c
#include<stdio.h>
#include<unistd.h>
int main(){
if (fork() && fork()){
fork();
}
printf("Hello\n");
return 0;
}
```



```
jay@jay-VirtualBox:~$ nano second.c
jay@jay-VirtualBox:~$ gcc second.c
jay@jay-VirtualBox:~$
```



```
jay@jay-VirtualBox:~$ nano second.c
jay@jay-VirtualBox:~$ gcc second.c
jay@jay-VirtualBox:~$ ./a.out
Hello
Hello
Hello
Hello
jay@jay-VirtualBox:~$
```