LINUX

What is Linux:

- Just like Windows, iOS, and Mac OS. Linux is an operating system. In fact, one of the most popular platforms on the planet, Android, is powered by the Linux operating system.
- An operating system is software that manages all of the hardware resources associated with your desktop or laptop.
- To put it simply, the operating system manages the communication between your software and your hardware.
- Without the operating system (OS), the software wouldn't function. Linux is open source.

How To Install Linux:

Method 1:

- Set Up Operating System To Linux (ubuntu Software) using VM Ware in System using both Windows & Linux.

Method 2:

- Update OS to Linux completely.

Method 3:(Installation Linux On Windows Without Changing Full Os)

- Using Online Server.Eg:Linuxzoo
- Install Putty From https://www.putty.org/ or https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html according to system configuration.
- Go to https://linuxzoo.net/ website and register.
- With help of IP Address ,Username and password connect to server in Putty.

Linux Directory Commands

1. pwd Command

The pwd command is used to display the location of the current working directory.

Syntax: pwd

Eg: [root@host-1-65 ~]# pwd

/root

2. mkdir Command

The mkdir command is used to create a new directory under any directory.

Syntax: mkdir **<directory** name**>**

Eg: [root@host-1-65 ~]# mkdir examplefolder

3. rmdir Command

The rmdir command is used to delete a directory.

Syntax: rmdir **<directory** name**>**

Eg: [root@host-1-65 ~]# rmdir examplefolder

4. ls Command

The ls command is used to display a list of content of a directory.

Syntax: ls

Eg: [root@host-1-65 ~]# ls

- **ls –l :** file or directory, size, modified date and time, file or folder name and owner of file and its permission.
- **ls** –a : List all files including hidden file starting with '**.**'
- **ls** –lh: shows sizes in human readable format.
- **ls** –F : will add the '**/**' Character at the end each directory.
- **ls -r** : display files and directories in reverse order.
- **ls –R :** list very long listing directory trees.
- **ls –ltr:** latest modification file or directory date as last.
- **ls –lS :** displays file size in order, will display big in size first.
- **ls -i** : file / directory with inode number.
- ls --version: version of ls command.
- ls –help: List help page of ls command.
-

5. cd Command

The cd command is used to change the current directory.

Syntax: cd **<directory** name**>**

Eg: [root@host-1-65 ~]# cd examplefolder

[root@host-1-65 examplefolder]#

Note:

cd .. : come out of current directory.

cd : come to home directory.

cd -: Go to Next directory.

Linux File Commands:

6. touch Command

The touch command is used to create empty files. We can create multiple empty files by executing it once.

Syntax:

touch **<file** name**>**
touch **<file1> <file2>** ....
Eg: [root@host-1-65 ~]# touch file.txt

7. cat Command

The cat command is a multi-purpose utility in the Linux system. It can be used to create a file, display content of the file, copy the content of one file to another file, and more.

Syntax: cat > filename
Eg: [root@host-1-65 ~]# cat>demo.txt
Hai hello welcome to linux
Happy learning.

It creates file and can write

Click Ctrl+D to save the file.

Syntax: cat filename
Eg: [root@host-1-65 ~]# cat demo.txt

It displays the content in given file

Syntax: cat>filename1 filename2
Eg: [root@host-1-65 ~]# cat>demo1.txt demo.txt

It copies the content of file2 into file1; if file1 is not present, it creates a new file an copy content from file2.

Syntax: cat filename1 filename2 > filename3

Eg: [root@host-1-65 ~]# cat demo1.txt demo.txt >new.txt

It copies the content of file1 and file2 into file3; if file3 is not present, it creates a new file and copy content from both files.

8. rm Command

The rm command is used to remove a file.

Syntax: rm <file name>

Eg:[root@host-1-65 ~]#rm demo.txt

[root@host-1-65 ~]#rm demo1.txt demo2.txt

9. cp Command

The cp command is used to copy a file or directory.

Syntax:

cp **<existing** file name**>** **<new** file name**>**

Eg:[root@host-1-65 ~]#cp demo.txt newfile.txt

To copy in the same directory:

To copy in a different directory:

Syntax:

cp <existing filename> <directory>

10. mv Command

The mv command is used to move a file or a directory form one location to another location.

Syntax:

mv **<file** name**>** **<directory** path**>**

Eg:

[root@host-1-65 ~]#mv demo.txt examplefolder

11.clear command:

It clears whole screen.

12. **grep Command:**

The grep is the most powerful and used filter in a Linux system. The 'grep' stands for "**global regular expression print**." It is useful for searching the content from a file. Generally, it is used with the pipe.

Syntax: command | grep **<searchWord>**

Eg: cat filename.txt | grep searchingword

13. sed command

The sed command is also known as **stream editor**. It is used to edit files using a regular expression. It does not permanently edit files; instead, the edited content remains only on display. It does not affect the actual file.

Syntax: command | sed 's/**<oldWord>**/**<newWord>**/'

Eg: cat filename | sed ‘s/h/7/’

14. tac Command

The tac command is the reverse of cat command, as its name specified. It displays the file content in reverse order (from the last line).

Syntax: tac <**file** name>

15. head Command

The head command is used to display the content of a file. It displays the first 10 lines of a file.

Syntax: head <**file** name>

16. tail Command

The tail command is similar to the head command. The difference between both commands is that it displays the last ten lines of the file content. It is useful for reading the error message.

Syntax: tail <**file** name>

17. more command

The more command is quite similar to the cat command, as it is used to display the file content in the same way that the cat command does. The only difference between both commands is that, in case of larger files, the more command displays screenful output at a time.

In more command, the following keys are used to scroll the page:

ENTER key: To scroll down page by line.

Space bar: To move to the next page.

b key: To move to the previous page.

/ key: To search the string.

Syntax: more <file name>

18. less Command

The less command is similar to the more command. It also includes some extra features such as 'adjustment in width and height of the terminal.' Comparatively, the more command cuts the output in the width of the terminal.

Syntax: less <file name>

Navigating Through the File Content:

- When opening a file which content is too large to fit in one page, you will see a **single colon (:)**
- To go forward to the next page press either the **f key** or **Space bar**. If you want to move down for a specific number of lines, type the number followed by the space or f key.
- You can press either the Down arrow or Enter to scroll forward by one line and Up arrow scroll backward by one line.
- To go back to the previous page hit the b key. Move up for a specific number of lines, by typing the number followed by the b key.
- If you want to search for a pattern, type forward slash (/) followed by the pattern you want to search. Once you hit Enter less will search forward for matches. To search backwards use (?) followed by the search pattern.
- When the end of the file is reached, the string (END) is shown at the bottom of the screen.
- To quit less and go back to the command line **press q.**

19. df command:

It displays the information of device name, total blocks, total disk space, used disk space, available disk space and mount points on a file system.

Syntax: df

- df –a : displays information of dummy file systems along with all the file system disk usage and their memory utilization.

- df –h: display sizes in **Human Readable** formats.
- df -hT /home : only device **/home** file system in human readable format
- df –k : To display all file system information and usage in **1024-byte** blocks
- df –m: To display information of all file system usage in **MB** (**Mega Byte**)
- df –i: display the information of number of used inodes and their percentage for the file system.
- df –T: display file system type along with other information.

20. echo command:

echo command in linux is used to display line of text/string that are passed as an argument .

Syntax :

echo [option] [string]

Ex: echo “String”

Options of echo command

NOTE :- -e here enables the interpretation of backslash escapes

1. \b : it removes all the spaces in between the text

Example :

echo -e "Geeks \bfor \bGeeks"

2. \c : suppress trailing new line with backspace interpretor ‘-e‘ to continue without emitting new line.

Example :

echo -e "Geeks \cfor Geeks"

In above example, text after \c is not printed and omitted trailing new line.

3. \n : this option creates new line from where it is used.

Example :

echo -e "Geeks \nfor \nGeeks"

4. \t : this option is used to create horizontal tab spaces.

Example :

echo -e "Geeks \tfor \tGeeks"

5. \r : carriage return with backspace interpretor ‘-e‘ to have specified carriage return in output.

Example :

echo -e "Geeks \rfor Geeks"

In the above example, text before \r is not printed.

6. \v : this option is used to create vertical tab spaces.

Example :

echo -e "Geeks \vfor \vGeeks

7. \a : alert return with backspace interpretor '-e' to have sound alert.

Example :

echo -e "\aGeeks for Geeks

This command when executed, it will produce an alert sound or Bel .

8. echo * : this command will print all files/folders, similar to ls command .

Example :

echo *

9. -n : this option is used to omit echoing trailing newline .

Example :

echo -n "Geeks for Geeks"

Linux File Permissions:

In Linux, each file is associated with an owner and a group and assigned with permission access rights for three different classes of users:

- The file owner.
- The group members.
- Others (everybody else).

File ownership can be changed using the `chown` and `chgrp` commands.

There are three file permissions types that apply to each class:

- The read permission.
- The write permission.
- The execute permission.

This concept allows you to specify which users are allowed to read the file, write to the file, or execute the file.

File permissions can be viewed using the `ls` command:

```
Ex:
ls -l filename.txt
```

Output:

```
-rw-r--r-- 12 linuxize users 12.0K Apr  8 20:51 filename.txt
|[-][-][-]-   [------] [---]
| |  |  | |       |       |
| |  |  | |       |       +-----------> 7. Group
| |  |  | |       +-------------------> 6. Owner
| |  |  | +---------------------------> 5. Alternate Access Method
| |  |  +-----------------------------> 4. Others Permissions
| |  +--------------------------------> 3. Group Permissions
| +-----------------------------------> 2. Owner Permissions
+-------------------------------------> 1. File Type
```

In the example above (`rw-r--r--`) means that the file owner has read and write permissions (`rw-`), the group and others have only read permissions (`r--`).

Effect of Permissions on Files

Permission	Character	Meaning on File
Read	`-`	The file is not readable. You cannot view the file contents.
	`r`	The file is readable.
Write	`-`	The file cannot be changed or modified.
	`w`	The file can be changed or modified.
Execute	`-`	The file cannot be executed.
	`x`	The file can be executed.
	`s`	If found in the `user` triplet it sets the `setuid` bit. If found in the `group` triplet, it sets the `setgid` bit. It also means that `x` flag is set. When the `setuid` or `setgid` flags are set on an executable file, the file is executed with the file's owner and/or group privileges.
	`S`	Same as `s` but the `x` flag is not set. This flag is rarely used on files.
	`t`	If found in the `others` triplet it sets the `sticky` bit. It also means that `x` flag is set. This flag is useless on files.
	`T`	Same as `t` but the `x` flag is not set. This flag is useless on files.

Effect of Permissions on Directories (Folders)

In Linux, Directories are special types of files that contain other files and directories.

Permission	Character	Meaning on Directory
Read	-	The directory's contents cannot be shown.
	r	The directory's contents can be shown. (e.g. You can list files inside the directory with `ls`.)
Write	-	The directory's contents cannot be altered.
	w	The directory's contents can be altered. (e.g. You can create new files, delete files ..etc.)
Execute	-	The directory cannot be changed to.
	x	The directory can be navigated using `cd`.
	s	If found in the `user` triplet, it sets the `setuid` bit. If found in the `group` triplet it sets the `setgid` bit. It also means that `x` flag is set. When the `setgid` flag is set on a directory the new files created within it inherits the directory group ID (GID), instead of the primary group ID of the user who created the file. `setuid` has no effect on directories.
	S	Same as `s` but the `x` flag is not set. This flag is useless on directories.
	t	If found in the `others` triplet it sets the `sticky` bit. It also means that `x` flag is set. When the sticky bit is set on a directory, only the file's owner, the directory's owner, or administrative user can delete or rename the files within the directory.
	T	Same as `t` but the `x` flag is not set. This flag is useless on directories.

Using `chmod`

The `chmod` command takes the following general form:

```
chmod [OPTIONS] MODE FILE...
```

The `chmod` command allows you to change the permissions on a file using either a symbolic or numeric mode or a reference file.

Symbolic (Text) Method

The syntax of the `chmod` command when using the symbolic mode has the following format:

```
chmod [OPTIONS] [ugoa…][-+=]perms…[,…] FILE...
```

The first set of flags (`[ugoa…]`), users flags, defines which users classes the permissions to the file are changed.

- `u` - The file owner.
- `g` - The users who are members of the group.
- `o` - All other users.
- `a` - All users, identical to `ugo`.

- If the users flag is omitted, the default one is `a` and the permissions that are set by umask are not affected.
  The second set of flags (`[-+=]`), the operation flags, defines whether the permissions are to be removed, added, or set:
  - Removes the specified permissions.
- `+` Adds specified permissions.
- `=` Changes the current permissions to the specified permissions. If no permissions are specified after the `=` symbol, all permissions from the specified user class are removed.

The permissions (`perms...`) can be explicitly set using either zero or one or more of the following letters: `r`, `w`, `x`, `X`, `s`, and `t`. Use a single letter from the set `u`, `g`, and `o` when copying permissions from one to another users class.

When setting permissions for more than one user classes (`[,…]`), use commas (without spaces) to separate the symbolic modes.

Below are some examples of how to use the `chmod` command in symbolic mode:

- Give the members of the group permission to read the file, but not to write and execute it:

```
chmod g=r filename
```

- Remove the execute permission for all users:

```
chmod a-x filename
```

- Repulsively remove the write permission for other users:

```
chmod -R o-w dirname
```

- Remove the read, write, and execute permission for all users except the file's owner:

```
chmod og-rwx filename
```

The same thing can be also accomplished by using the following form:

```
chmod og= filename
```

- Give read, write and execute permission to the file's owner, read permissions to the file's group and no permissions to all other users:

```
chmod u=rwx,g=r,o= filename
```

- Add the file's owner permissions to the permissions that the members of the file's group have:

```
chmod g+u filename
```

- Add a sticky bit to a given directory:

```
chmod o+t dirname
```

Numeric Method

The syntax of the `chmod` command when using numeric method has the following format:

```
chmod [OPTIONS] NUMBER FILE...
```

When using the numeric mode, you can set the permissions for all three user classes (owner, group, and all others) at the same time.

The `NUMBER` can be a 3 or 4-digits number.

When 3 digits number is used, the first digit represents the permissions of the file's owner, the second one the file's group, and the last one all other users.

Each write, read, and execute permissions have the following number value:

- `r` (read) = 4
- `w` (write) = 2
- `x` (execute) = 1
- no permissions = 0

The permissions number of a specific user class is represented by the sum of the values of the permissions for that group.
To find out the file's permissions in numeric mode simply calculate the totals for all users classes. For example, to give read, write and execute permission to the file's owner, read and execute permissions to the file's group and only read permissions to all other users you would do the following:

- Owner: rwx=4+2+1=7
- Group: r-x=4+0+1=5
- Others: r-x=4+0+0=4

Using the method above we come up to the number `754`, which represents the desired permissions.

To set up the `setuid`, `setgid`, and `sticky bit` flags use four digits number.

When the 4 digits number is used, the first digit has the following meaning:

- setuid=4
- setgid=2
- sticky=1
- no changes = 0

The next three digits have the same meaning as when using 3 digits number.

If the first digit is 0 it can be omitted, and the mode can be represented with 3 digits. The numeric mode `0755` is the same as `755`.

To calculate the numeric mode you can also use another method (binary method), but it is a little more complicated. Knowing how to calculate the numeric mode using 4, 2, and 1 is sufficient for most users.

You can check the file's permissions in the numeric notation using the `stat` command:

```
stat -c "%a" filename
644
```

Here are some examples of how to use the `chmod` command in numeric mode:

- Give the file's owner read and write permissions and only read permissions to group members and all other users:

```
chmod 644 dirname
```

- Give the file's owner read, write and execute permissions, read and execute permissions to group members and no permissions to all other users:

```
chmod 750 dirname
```

- Give read, write, and execute permissions, and a sticky bit to a given directory:

```
chmod 1777 dirname
```

- Recursively set read, write, and execute permissions to the file owner and no permissions for all other users on a given directory:

```
chmod -R 700 dirname
```

https://linuxize.com/post/chmod-command-in-linux/ -Refer for chmod

Opening A File:

1.using cat command

2.using vi command

3.using nano command (Recommended)

	cat	vi	nano
open	cat filename	vi filename	nano filename
save	Ctrl+D	:w	Ctrl+O
exit	-	:q	ctrl+X
Save & exit		:wq	

VI Editing commands

- i - Insert at cursor (goes into insert mode)
- a - Write after cursor (goes into insert mode)
- A - Write at the end of line (goes into insert mode)

- ESC - Terminate insert mode
- u - Undo last change
- U - Undo all changes to the entire line
- o - Open a new line (goes into insert mode)
- dd - Delete line
- 3dd - Delete 3 lines.
- D - Delete contents of line after the cursor
- C - Delete contents of a line after the cursor and insert new text. Press ESC key to end insertion.
- dw - Delete word
- 4dw - Delete 4 words
- cw - Change word
- x - Delete character at the cursor
- r - Replace character
- R - Overwrite characters from cursor onward
- s - Substitute one character under cursor continue to insert
- S - Substitute entire line and begin to insert at the beginning of the line
- ~ - Change case of individual character

Saving and Closing the file

- Shift+zz - Save the file and quit
- :w - Save the file but keep it open
- :q - Quit without saving
- :wq - Save the file and quit

Linux Introduction To Users:

It also tells how to create a second user account and run program on that with the help of su and sudo command.

17. su Command

The su command provides administrative access to another user. In other words, it allows access of the Linux shell to another user.

Syntax: su **<user** name**>**

1. whoami

It tells you about the system's username.

Syntax: whoami

2. who

The who command gives the information about the users logged on to the system.

Syntax: who

3. who am i

This command displays the information about the current user only.

Syntax: who am i

4. w

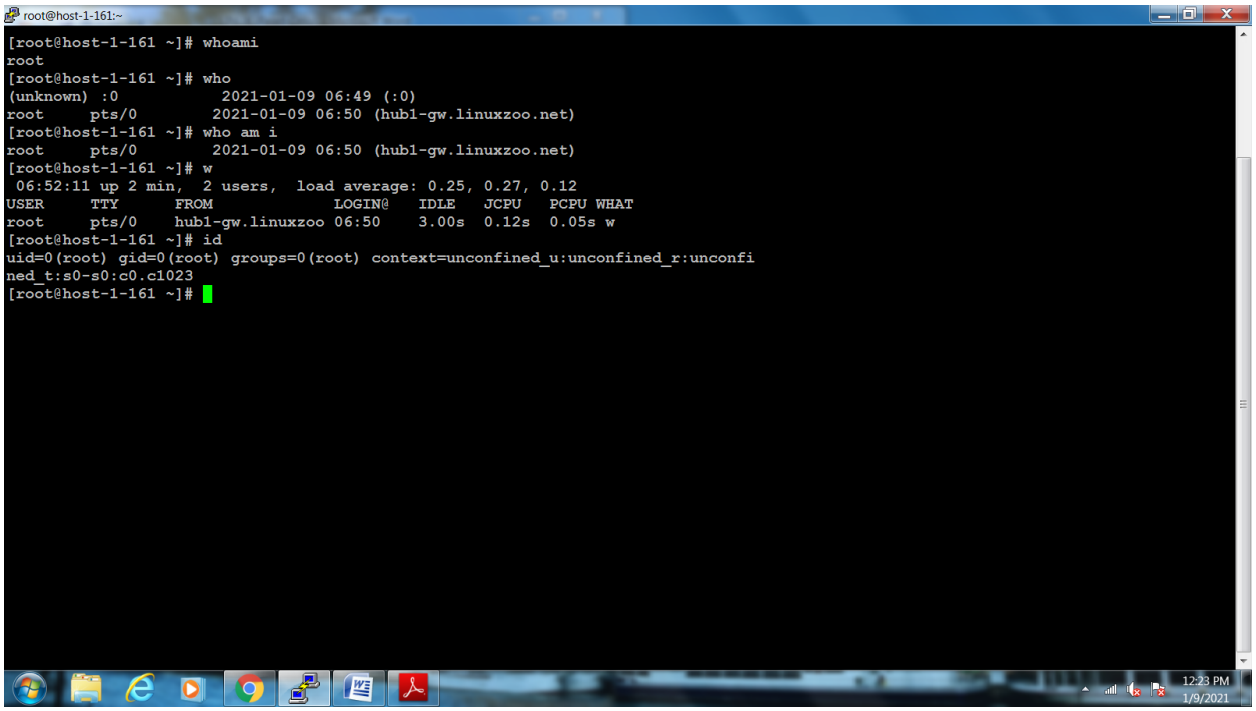
This command tells about the users who are logged in and what are they doing.

Syntax: w

5. id

This command tells about your user id, primary group id, and a list of groups that belongs to you.

Syntax: id



Create User in Linux (Ubuntu)

There are two most common ways to add a user to a Linux server.

- **Graphically through user manager**
- **By the useradd command (Terminal)**

1.By Graphically through user manager:

Linux GUI allows us to create a user from its functions. It is a straight forward process. To create a user to your Linux server, follow the below steps:

Step1: Goto system search and search for the **setting** and navigate to **Detail-> About.**

Step2: Click on the **Users** after that **Unlock option** given on the header. It will ask for the system security password to enter the password and click **ok** to continue.

Step3: Click on the **Add User** option to add a new user.

Step4: Enter the user details like username and password and account type. We can create two types of accounts, which are Standard and Administrator. The standard account does not contain the sudo privilege. However, we can provide it later.

Step5: Now, we have successfully created a new user called JTP2.

2. By the Linux useradd command

In Linux, **useradd command** is a command-line utility which is used to add or remove a user on a Linux server and Unix based operating system.

Syntax: useradd [options] username

useradd command in Linux with Examples

useradd is a command in Linux that is used to add user accounts to your system. It is just a symbolic link to adduser command in Linux and the difference between both of them is that useradd is a native binary compiled with system whereas adduser is a Perl script which uses useradd binary in the background. It make changes to the following files:

- /etc/passwd
- /etc/shadow
- /etc/group
- /etc/gshadow
- creates a directory for new user in /home

Syntax:

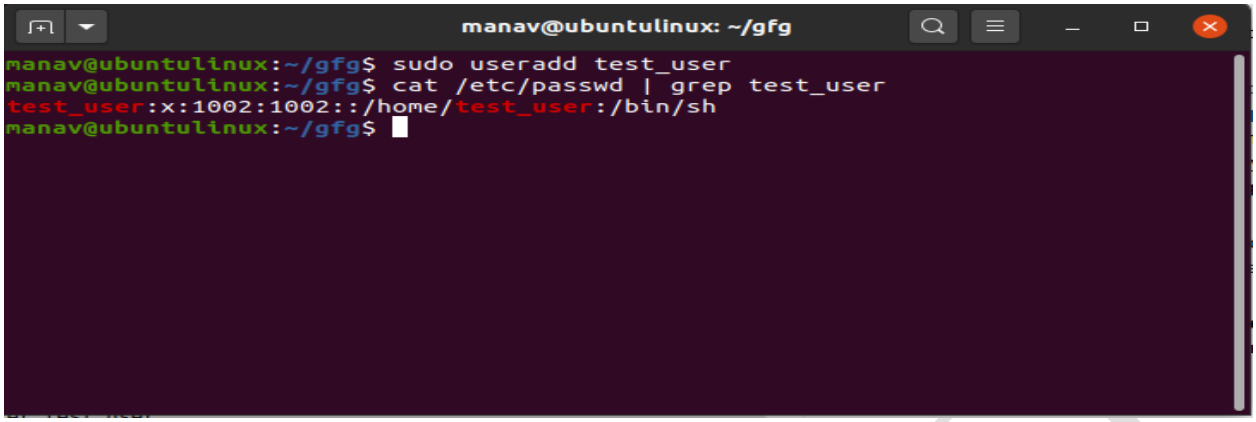
useradd [options] name_of_the_user

Working with useradd Command

1. To add a simple user

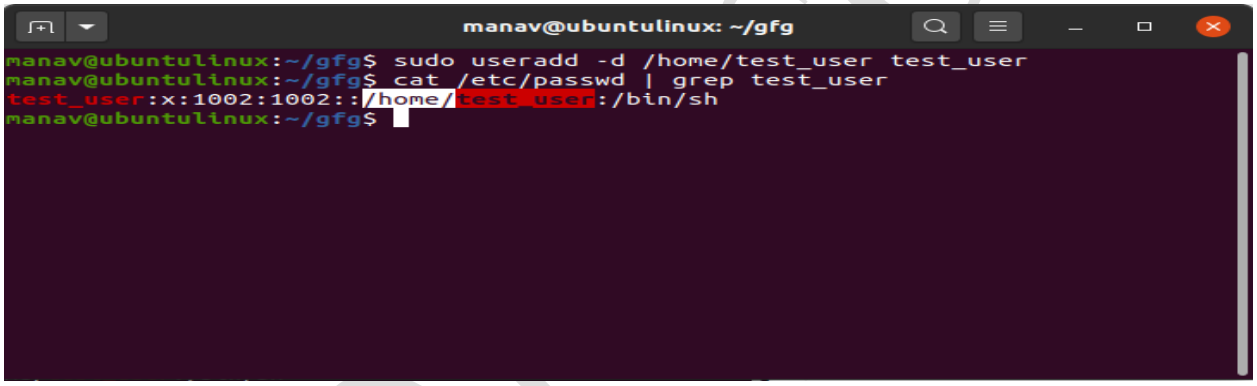
Syntax : sudo useradd test_user

This command will add the user named “test_user”.



2. To give a home directory path for new user

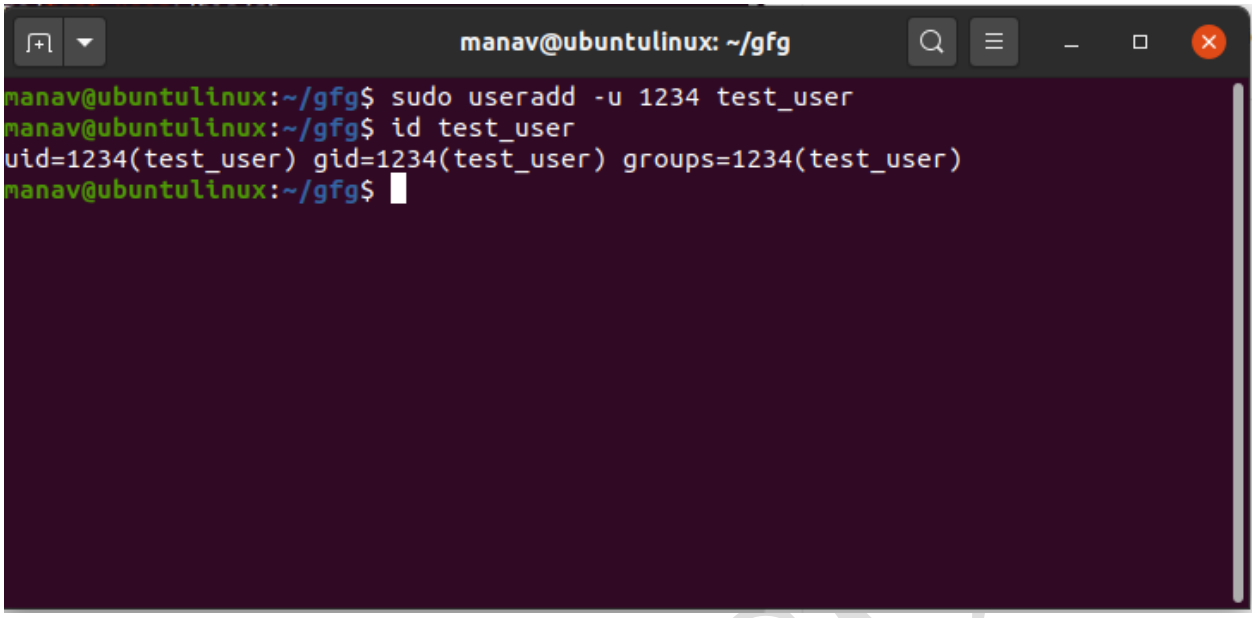
sudo useradd -d /home/test_user test_user



This will set the home directory of the us"/home/test_user".

3. To create a user with specific user id

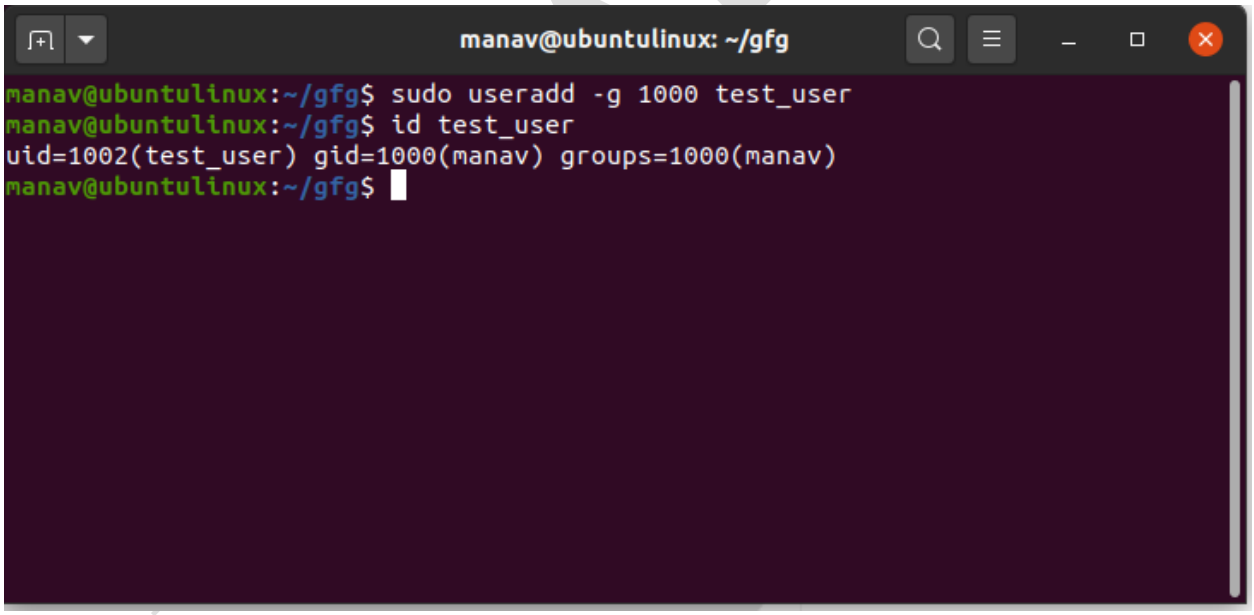
sudo useradd -u 1234 test_user



This will create a new user with the user-id “1234” and the name “test_user”.

4. To create a user with specific group id

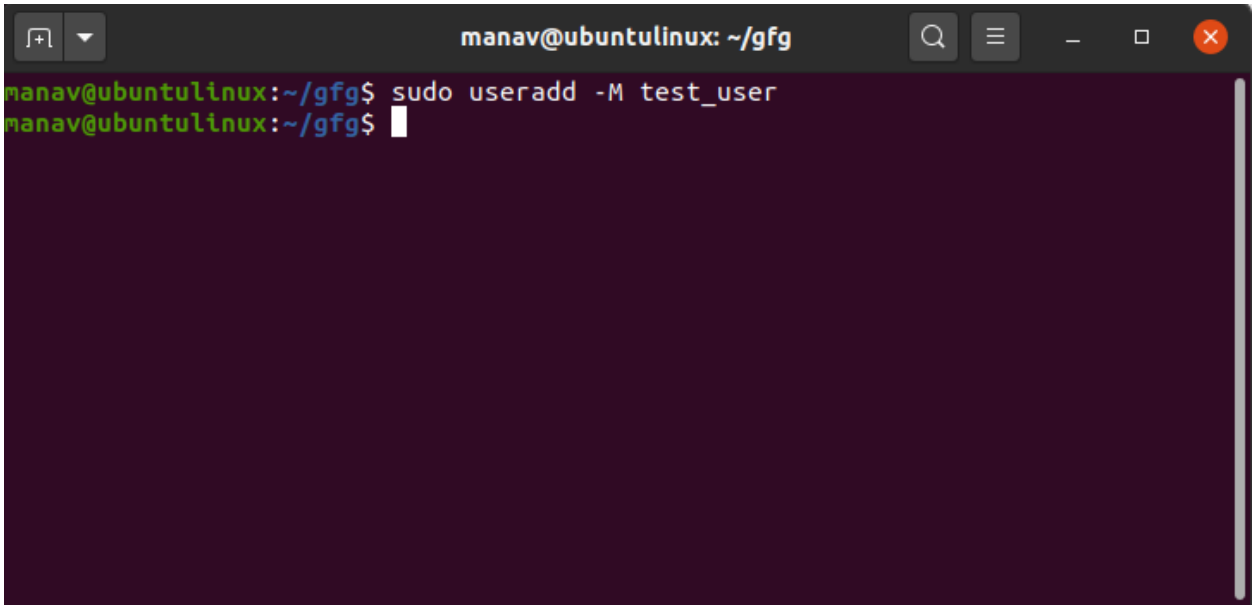
sudo useradd -g 1000 test_user



This will create a new user with the group id “1000” and the name “test_user”.

5. To create a user without home directory

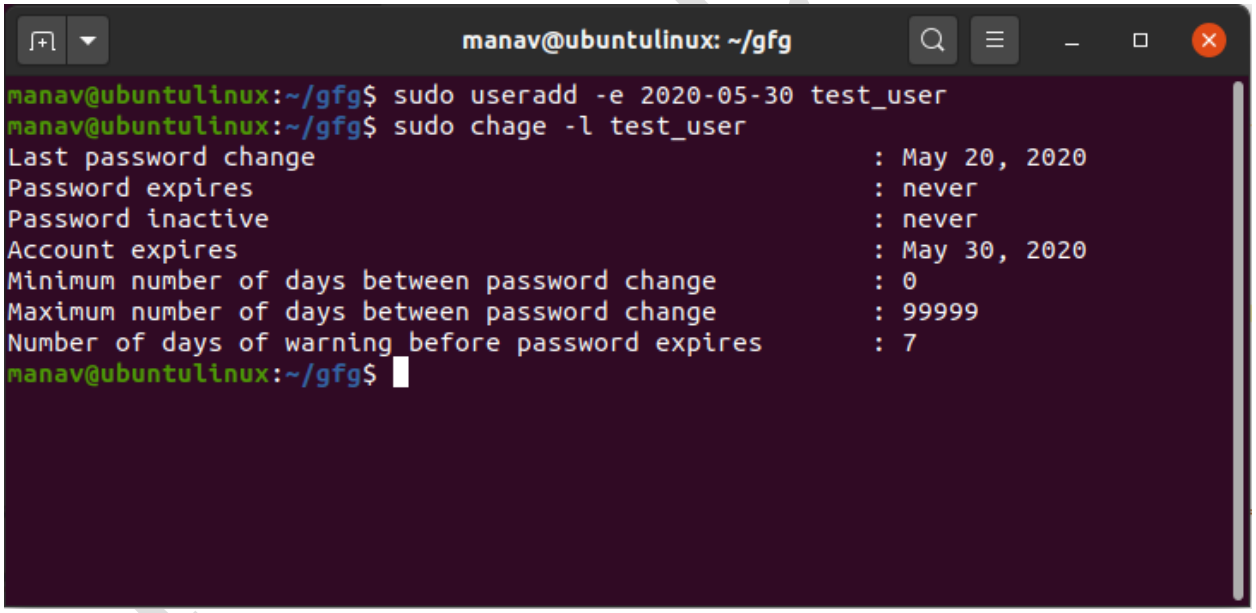
sudo useradd -M test_user



This will create the user with the name “test_user” and that too without a home directory.

6. To create a user with expiry date

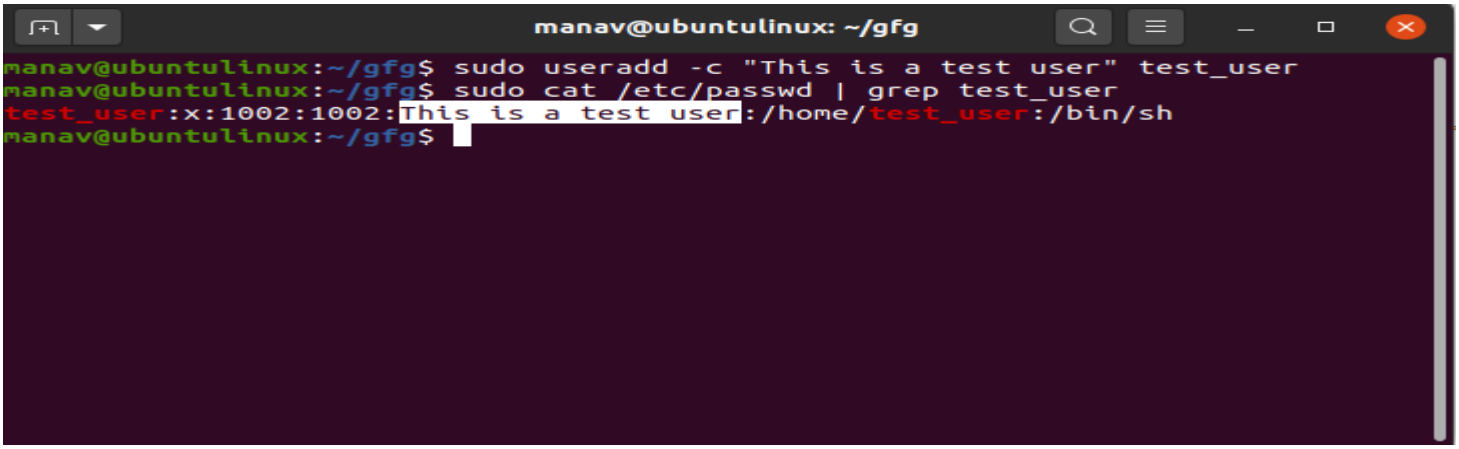
sudo useradd -e 2020-05-30 test_user



This will create the user named “test_user” with the expiry date of 30th May 2020.

7. To create a user with a comment

sudo useradd -c "This is a test user" test_user



This will create a user with a short comment or description of the user.

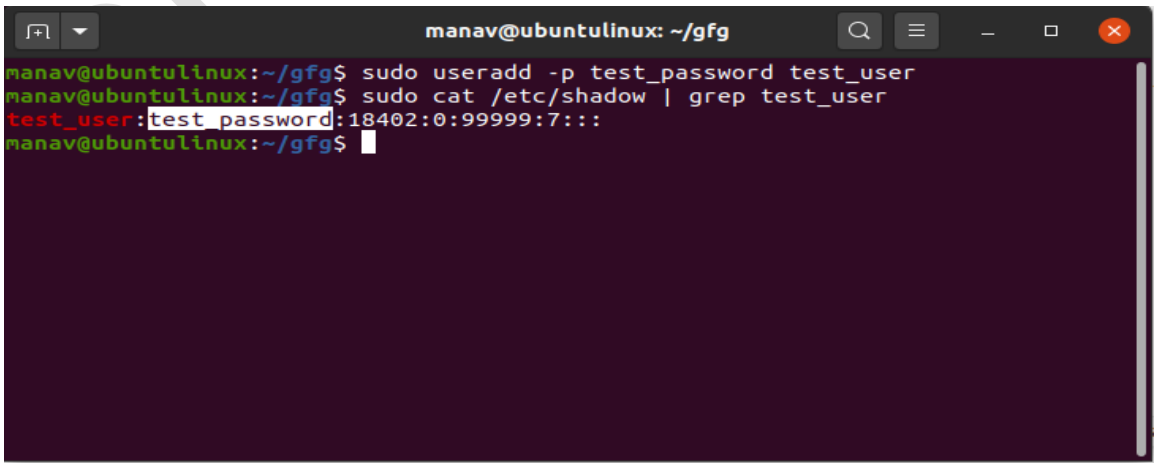
8. To create a user with changed login shell

sudo useradd -s /bin/sh test_user

This will create a user named “test_user” with the default shell /bin/sh.

9 To set an unencrypted password for the user

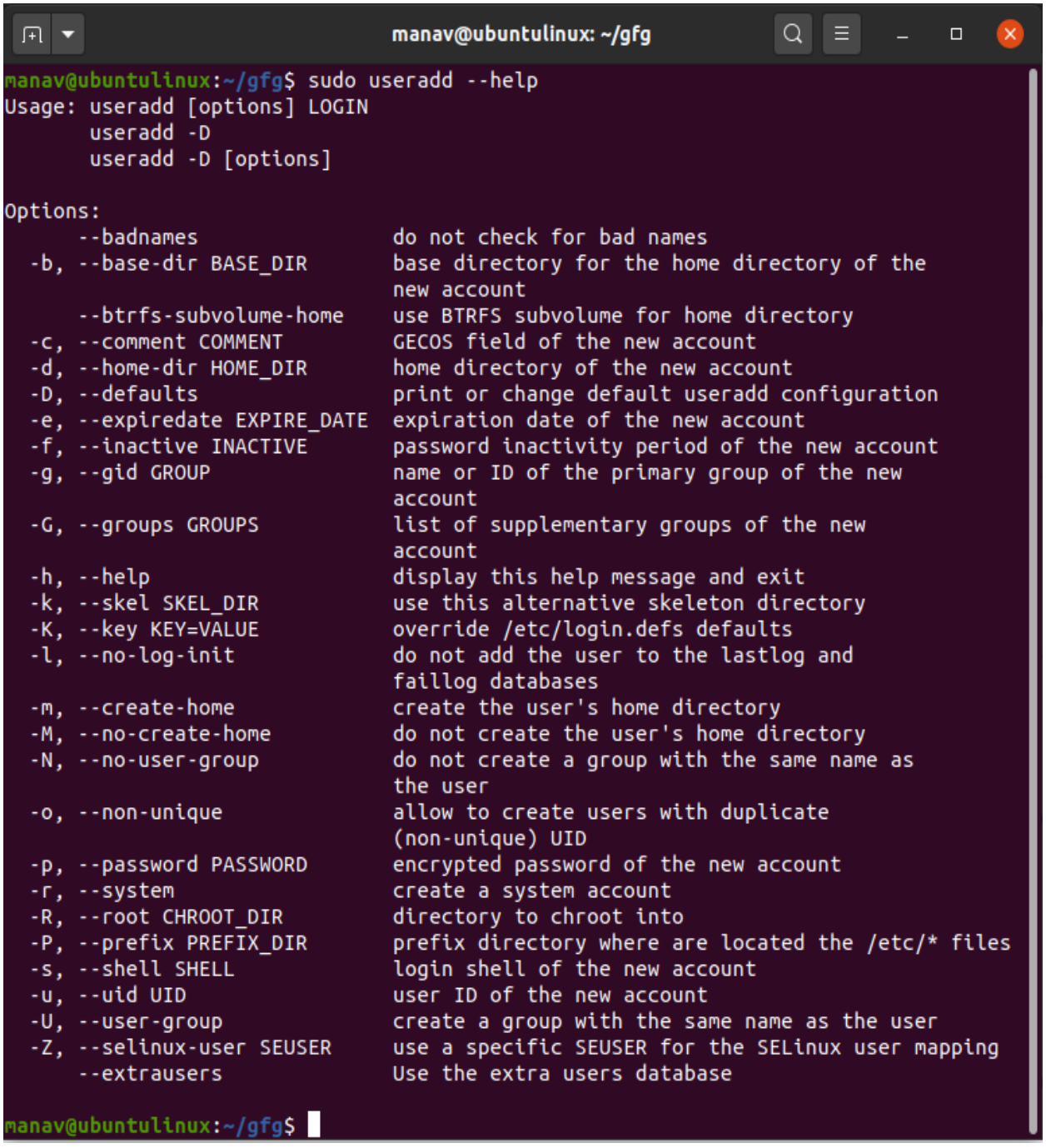
sudo useradd -p test_password test_user



This will create a new user with the name “test_user” and an unencrypted password “test_password”.

10. To display help

sudo useradd --help



This command will display the help section of the useradd command.