|  |
| --- |
| **Access To Linux System** |

There are two different ways, by which we can access a Linux System.

Accessing Linux Systems

Console Access

Remote Access

* Direct access to a Linux device via HDMI, DVI or VGA cables i.e a physical computer
* Connect to a Linux device remotely over the network

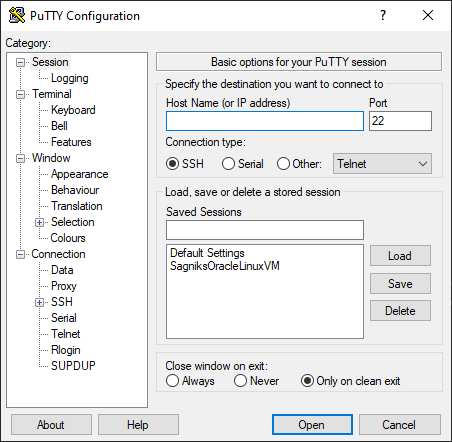
* **Ways To Remotely Connect To A Linux System**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl No** | **Source Machine** | **Destination Machine** | **Remote Connection Method** |
| 1 | Windows | Windows | Remote Desktop Client |
| 2 | Windows | Linux | Putty Client, SSH |
| 3 | Linux | Windows | SSH |
| 4 | Linux | Linux | SSH |

* **Putty Client ( Connecting To A Linux System From A Windows Machine)**

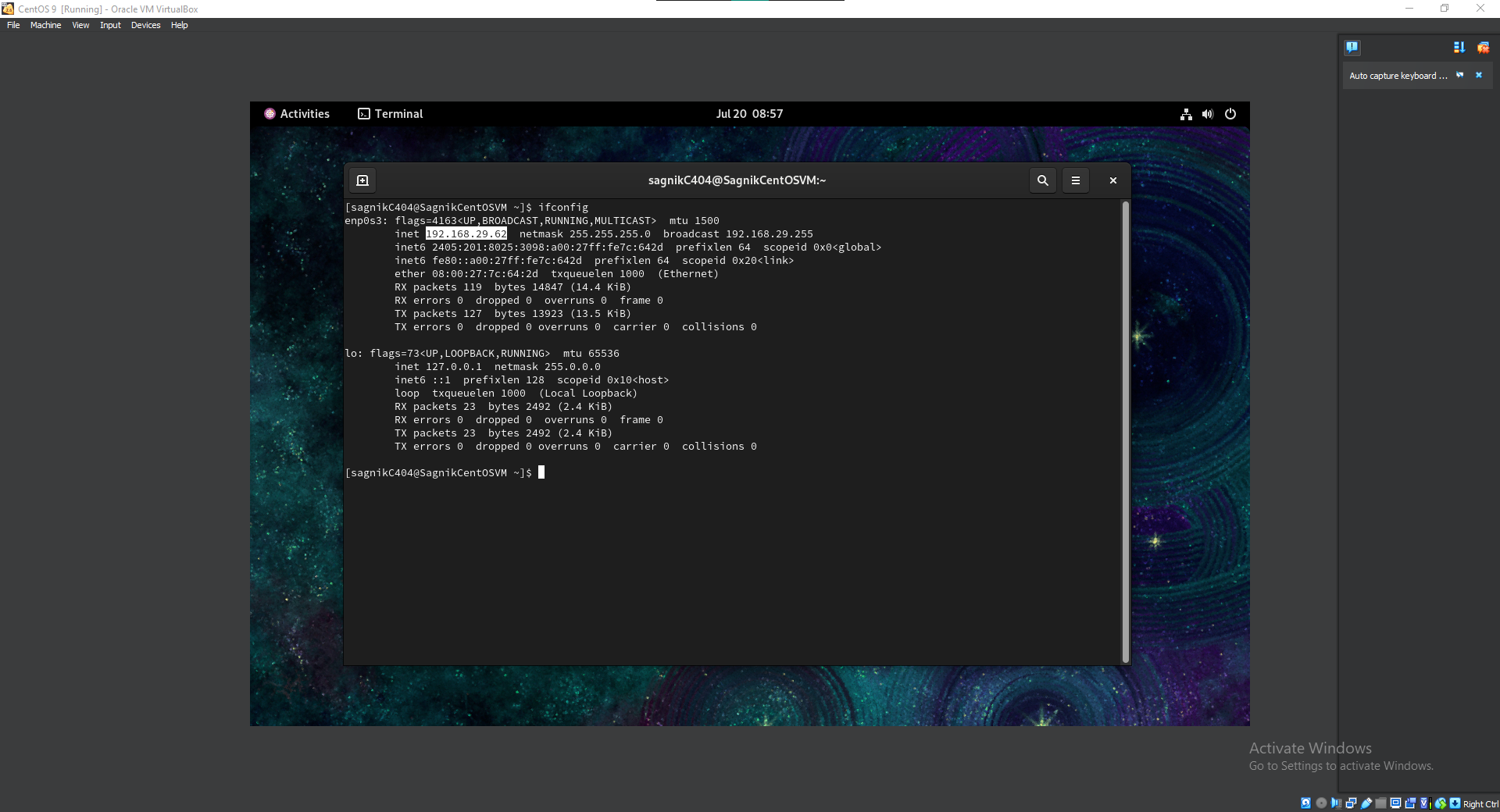
**Downloading Putty Client**

* First we need to download the Putty client from the official website using the following [link](https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html)
* Next we will need to install the program.
* After installation we need to open Putty



**Extracting The IP Address From The Linux VM**

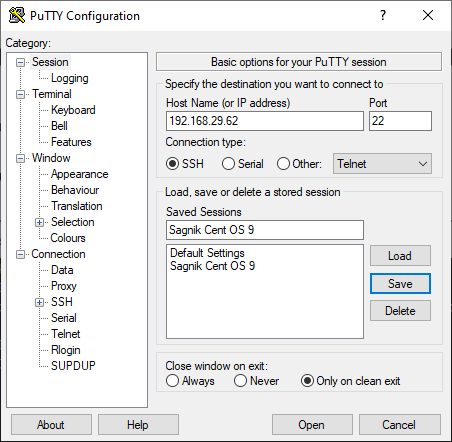
* First we need to open our VM
* Next we need to open up the terminal and type the following command
* ifconfig or ip addr



* As we can see in our output, the IP address of our VM Is 192.168.29.62.

**Connecting To The VM from Putty**

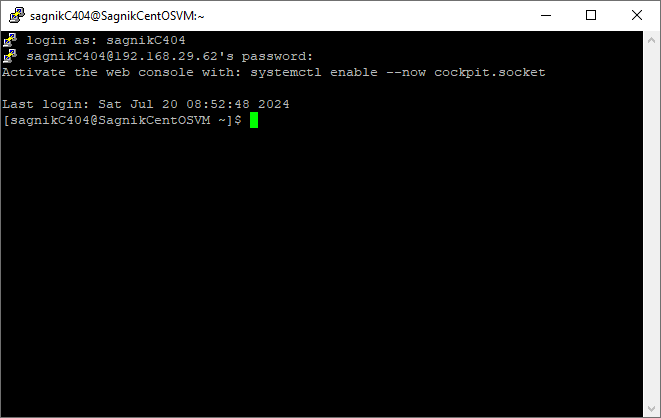
* First we need to open our Putty Client and fill in the extracted IP address and assign a name to the session
* Click on save.



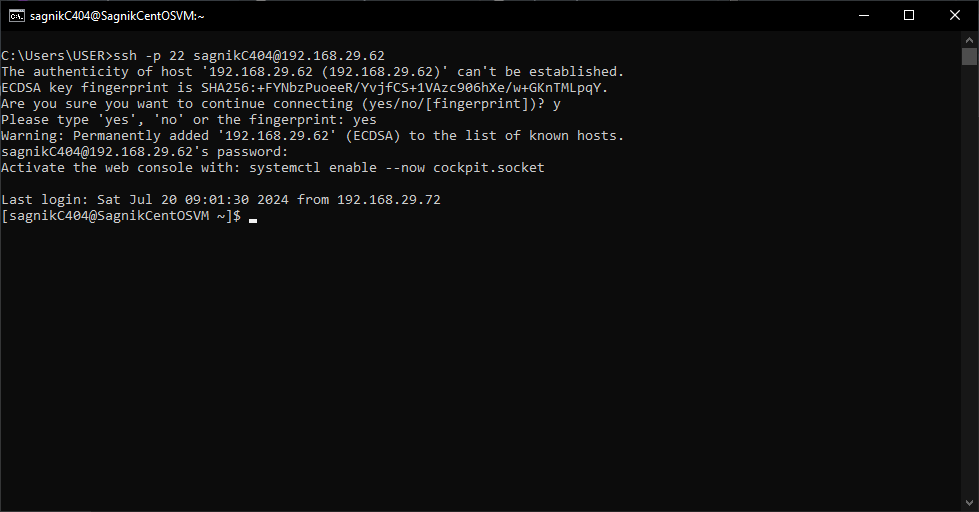
* Next we will need to click on open and we will see the connection establishing window which will ask for username and password



* Next we will need to fill in the username (Created During VM Set Up) followed by the password, which will be invisible.
* Hence we need to be precise while filling in the password.
* On successful authentication, we will see the following window



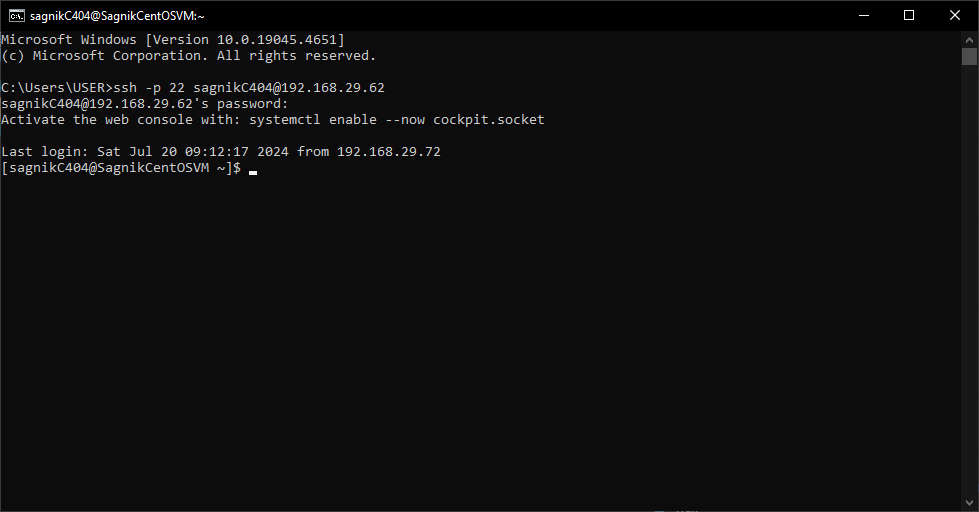
* We are ready to work in our VM.
* **SSH Client ( Connecting To A Linux System From A Windows Machine)**
* First we need to open our Console
* Next we need to type in the following command
* ssh -p 22 username@ipAddress
* It will ask for a confirmation. We have to write yes.
* Next it will ask for the password
* On successful authentication, we will see the following window



* SSH works only for Windows 10 and later version OS
* If we have anything below Windows 10, we will have to depend on Putty

|  |
| --- |
| **Prompts And How To Get Them Back** |

A command prompt, which is also simply referred to as a prompt is a short text which is seen at the beginning of the command line followed by a prompt symbol. It shows that the system is ready to take commands from us.

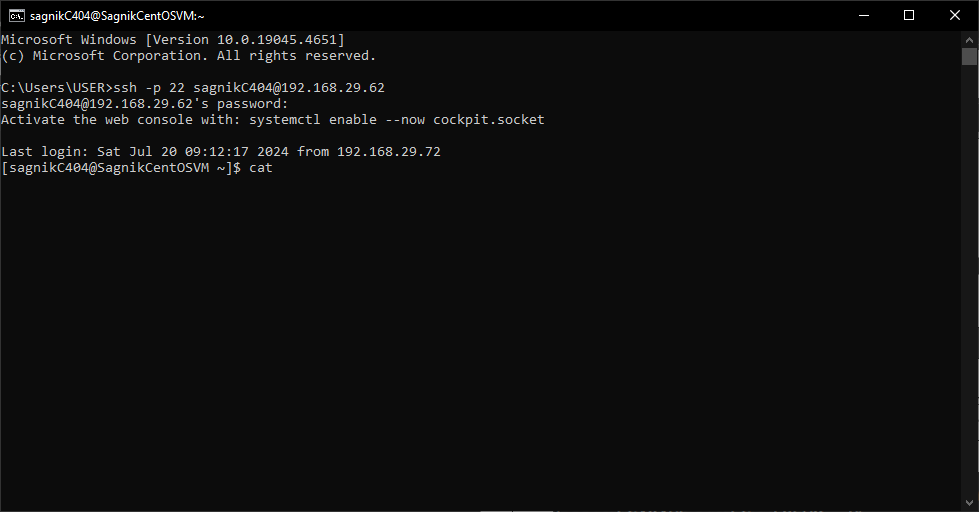


Username

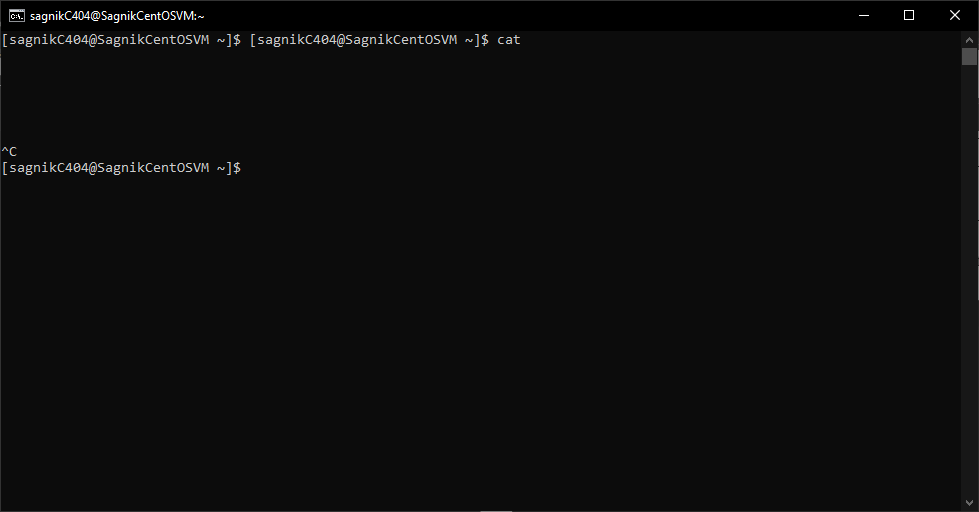
Hostname

Prompt Symbol

In case, while executing some operation, our prompt gets stuck and we feel like terminating the current process, we can hit Ctrl + C to get our prompt back



As we can see, we put in an invalid command and it is running endlessly. To get our prompt back, we can simply use Ctrl + C and we will see it come back



|  |
| --- |
| **Introduction To Linux File System** |

* **What is a file system?**

A file system is used by an operating system to manage files. The system controls how to save and retrieve files. File system stores files and directories in an organised manner.

Some examples of file systems are as follows

|  |  |  |
| --- | --- | --- |
| **Sl No** | **File System** | **Operating System** |
| 1 | ext3 | Linux |
| 2 | ext4 | Linux |
| 3 | xfs | Linux |
| 4 | NTFS | Windows |
| 5 | FAT | Windows |

* **File System Detailed Structure**

/

/bin

/sbin

/dev

/etc

/proc

/var

/tmp

/usr

/home

/boot

/root

/lib

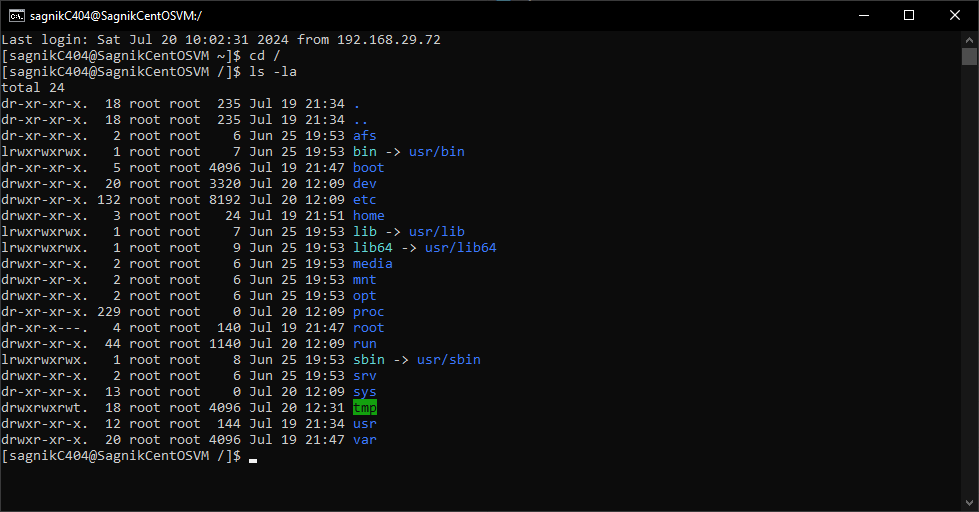
/opt

/mnt

/media

/srv v

If we cd into the root (/) directory and list out the contents over there, we will find all the above mentioned directories

****

* **Purpose Of The Directories**

|  |  |  |
| --- | --- | --- |
| **Sl No** | **Directory** | **Purpose** |
| 1 | / | Every single file and directory begins from this directory. Only root user has privilege to write under this directory |
| 2 | /bin | Contains binary executables. Common linux commands you need to use in single-user modes are located under this directory.  Commands used by all the users of the system are located here. For example: ps, ls, ping, grep, cp. |
| 3 | /sbin | System Binaries. Just like /bin, /sbin also contains binary executables. But, the linux commands located under this directory are used typically by system administrator, for system maintenance purpose.  For example: iptables, reboot, fdisk, ifconfig, swapon |
| 4 | /etc | Contains configuration files required by all programs. • This also contains startup and shutdown shell scripts used to start/stop individual programs.  For example: /etc/resolv.conf, /etc/logrotate.conf |
| 5 | /proc | Contains information about system process. This is a pseudo filesystem contains information about running process.  For example: /proc/{pid} directory contains information about the process with that particular pid. This is a virtual filesystem with text information about system resources. For example: /proc/uptime |
| 6 | /var | var stands for variable files. Content of the files that are expected to grow can be found under this directory.  This includes — system log files (/var/log); packages and database files (/var/lib); emails (/var/mail); print queues (/var/spool); lock files (/var/lock); temp files needed across reboots (/var/tmp); |
| 7 | /tmp | Directory that contains temporary files created by system and users. Files under this directory are deleted when system is rebooted. |
| 8 | /usr | Contains binaries, libraries, documentation, and source-code for second level programs. /usr/bin contains binary files for user programs.  If you can’t find a user binary under /bin, look under /usr/bin.  For example: at, awk, cc, less, scp /usr/sbin contains binary files for system administrators.  If you can’t find a system binary under /sbin, look under /usr/sbin.  For example: atd, cron, sshd, useradd, userdel /usr/lib contains libraries for /usr/bin and /usr/sbin  /usr/local contains users programs that you install from source. For example, when you install apache from source, it goes under /usr/local/apache2 |
| 9 | /home | Home directory for all users for storing their files  /home/john /home/nikita |
| 10 | /boot | Bootloader files. Kernel initrd, vmlinux, grub files are located under /boot |
| 11 | /lib | System libraries. Contains library files that support libraries located under /sbin and /bin |
| 12 | /opt | Optional add on files. Contains add on applications from individual vendors |
| 13 | /media | Removable media device. Temporary mount directory for removable devices.  For examples, /media/cdrom for CD-ROM; /media/floppy for floppy drives; /media/cdrecorder for CD writer |
| 14 | /mnt | Temporary mount directory where sysadmins can mount filesystems. |
| 15 | /srv | srv stands for service. Contains server specific services related data.  For example, /srv/cvs contains CVS related data |
| 16 | /root | Home directory for the root user |

|  |
| --- |
| **File System Navigation Command** |

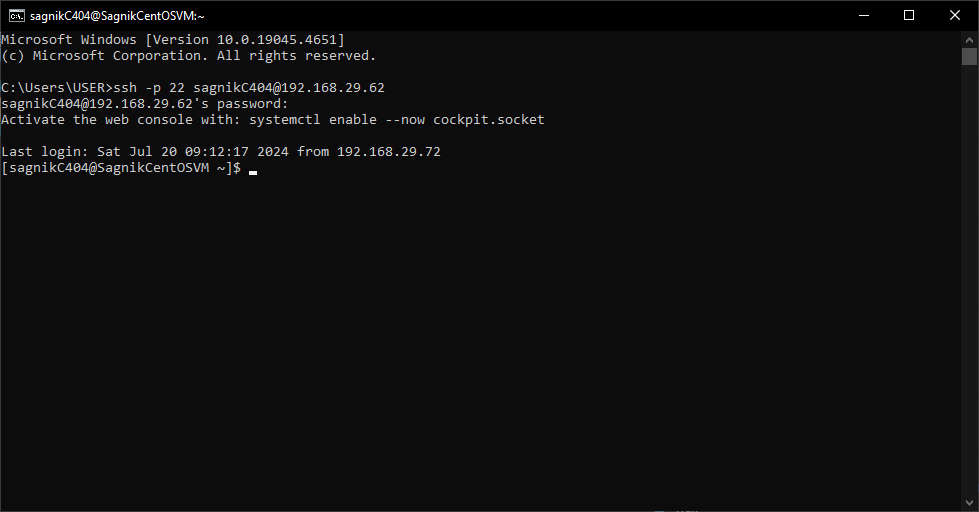
While navigating the Linux file system, we have three commands,

* pwd
* ls
* cd

|  |  |  |
| --- | --- | --- |
| **Sl No** | **Command** | **Purpose** |
| 1 | cd | cd stands for change directory |
| 2 | pwd | pwd stands for print working directory |
| 3 | ls | ls stands for list. It lists all files and directories present in a given directory |

* **Switching Between Normal And Root User**

We can take a look at the prompt and figure out, which user we are logged in as.

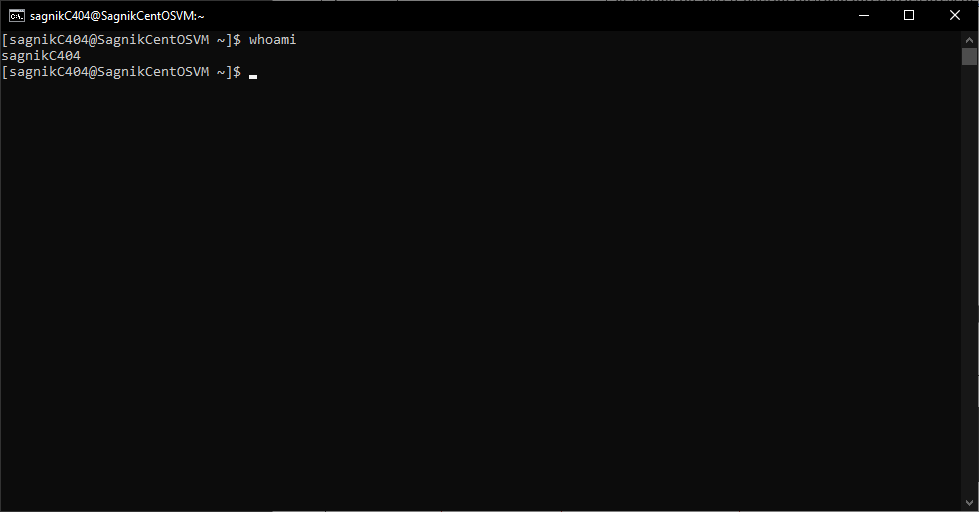


Username

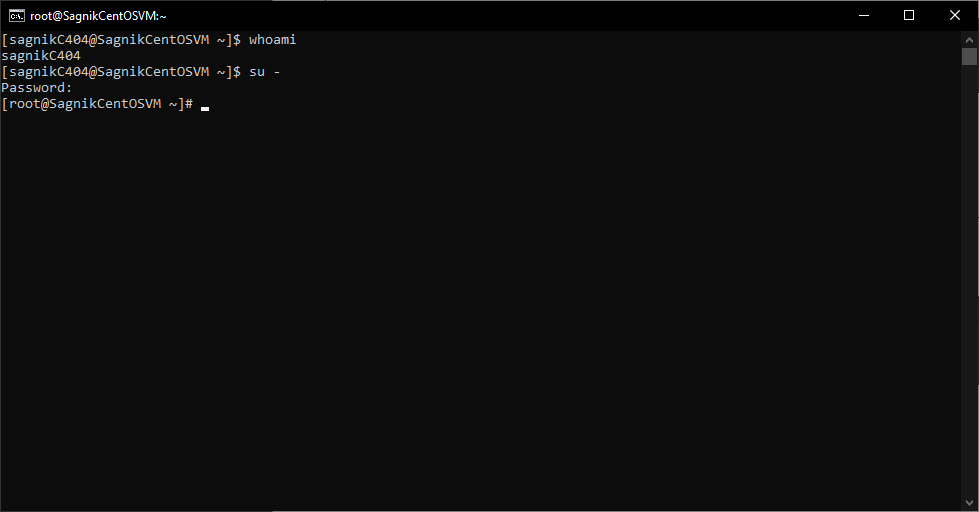
Hostname

Prompt Symbol

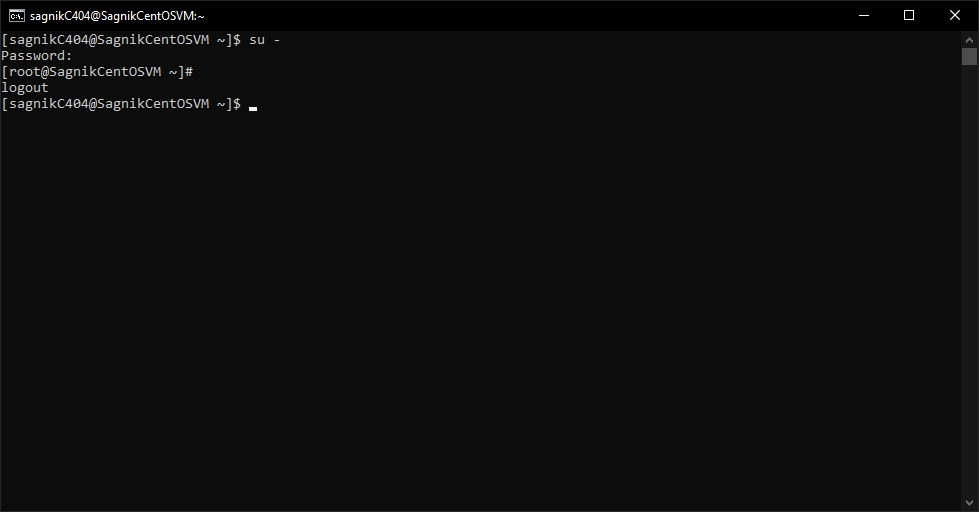
Also we can use the command whoami to show the username we are logged in as.



To switch to the root user, we can use the command su- . Next it will ask for the root user password. If the authentication is successful, we will be elevated as the root user.

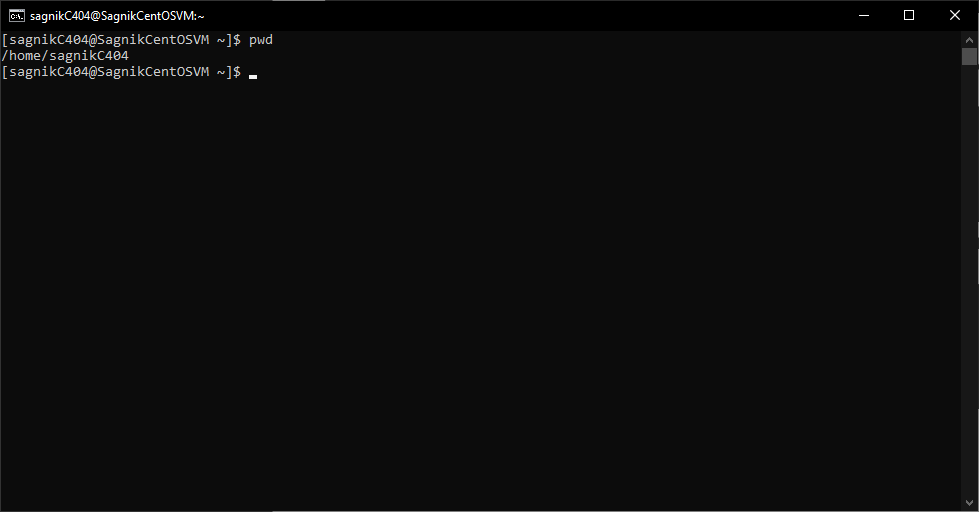


To relegate ourselves to a normal user from the root user mode, we can use Ctrl + D. We will be switched back to the normal user.

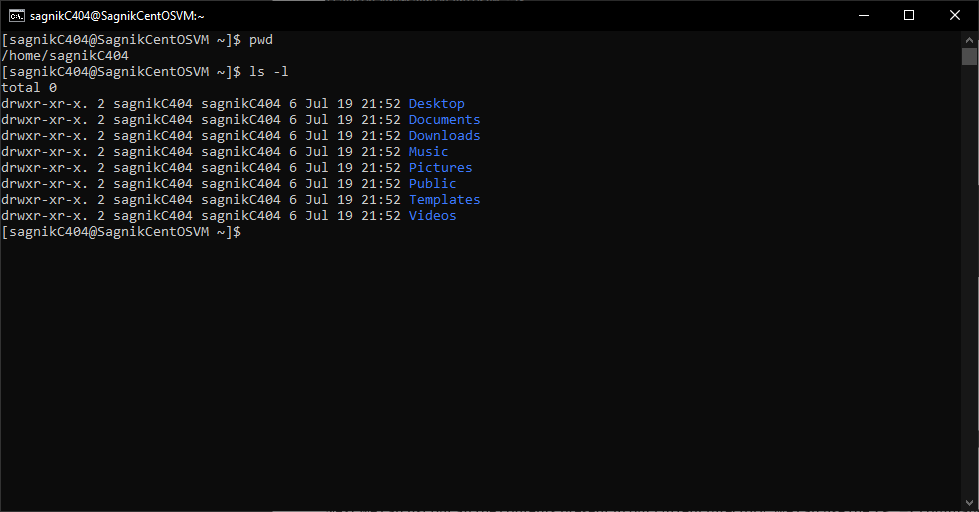


* **Using The pwd, cd and ls Command**

First let us use the pwd command to figure out the location we are currently at.

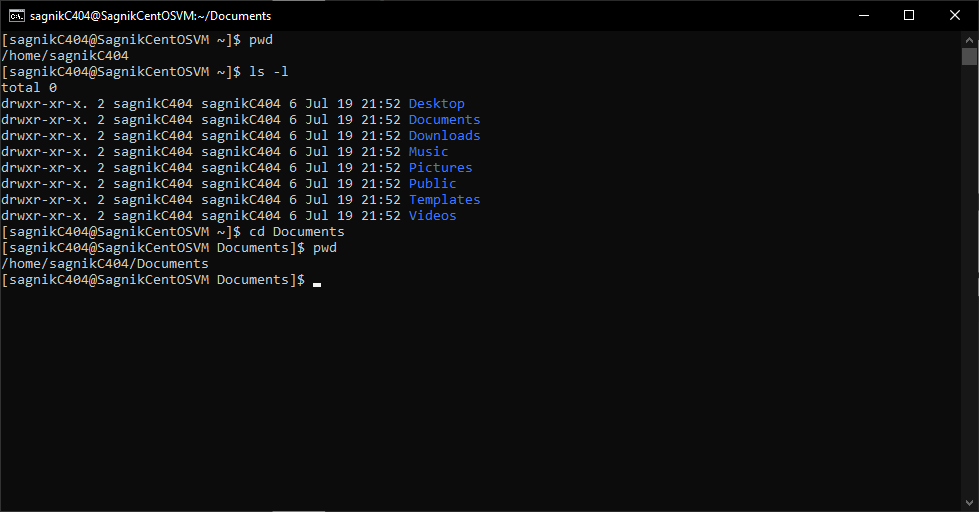


Next we can list out all the contents present in our current directory. We can use the ls -l command

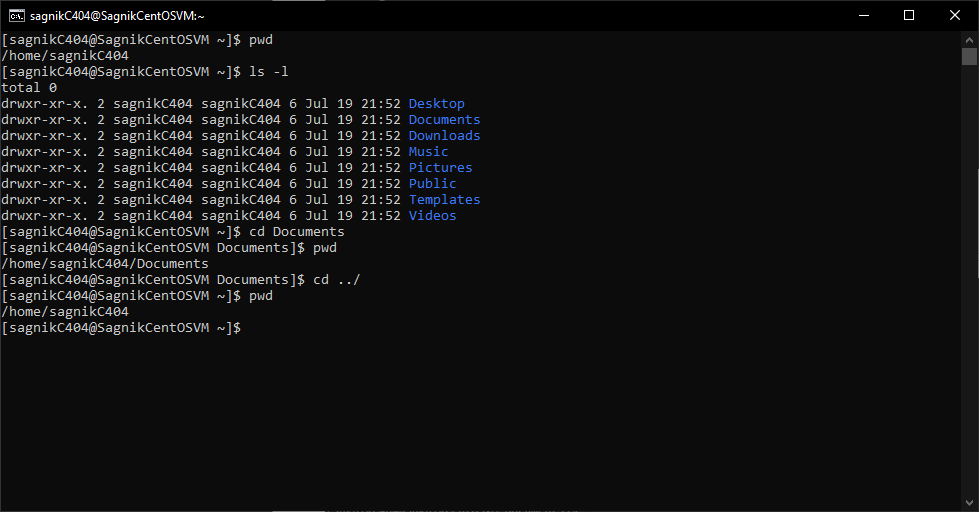


Next if we want to move into the Documents directory, we can use the command cd Documents.

To check if we were able to change the directory or not, we can check our location using the pwd command

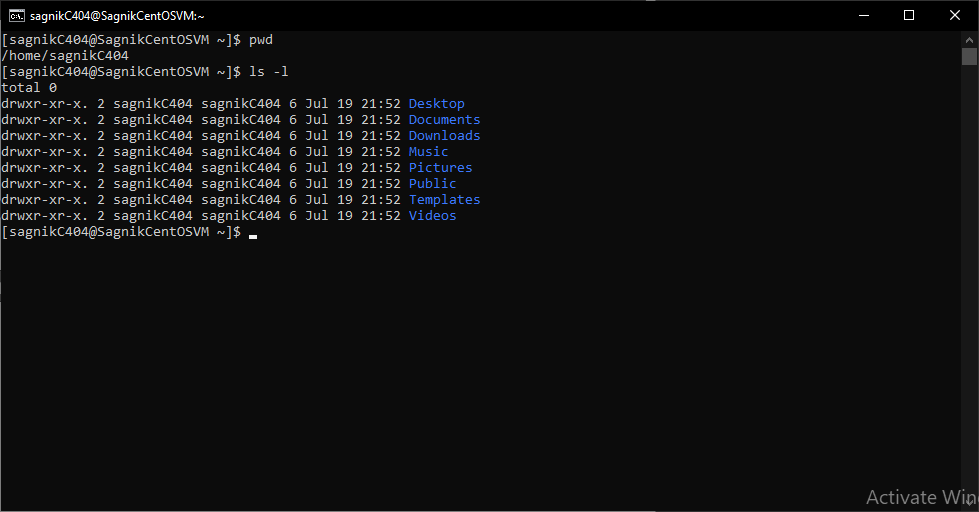


To comeback to the original user home directory we can use the cd ../



|  |
| --- |
| **Linux File Or Directory Properties** |

If we take a look at listing of the contents of a directory, we will see a lot of information being shown to us. Which are the properties of these files/directories



Let us take an example and analyse it

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Type Of File | No Of Links | Owner | Group | Size | Month | Day | Time | Name |
| drwxr-xr-x | 2 | sagnikC404 | sagnikC404 | 6 | Jul | 19 | 21:52 | Desktop |
| drwxr-xr-x | 2 | sagnikC404 | sagnikC404 | 6 | Jul | 19 | 21:52 | Documents |

* The first column tells us about the type of the file and the associated file permissions. Anything that starts with a ‘d’ is a directory. Anything that starts with a ‘l’ it is a link. If it has nothing, (-) it is a normal text file.
* The second column tells us about the number of hard links it has. It is the total number of parent directories and the sub directories it has.
* The third column tells us about the owner.
* The fourth column tells us about the group it belongs to
* The fifth column tells us about the size of the file.
* The sixth , seventh and eighth column tells us about the month, the date and the time when the file was created
* The ninth and the final column tells us the name of the file or directory.

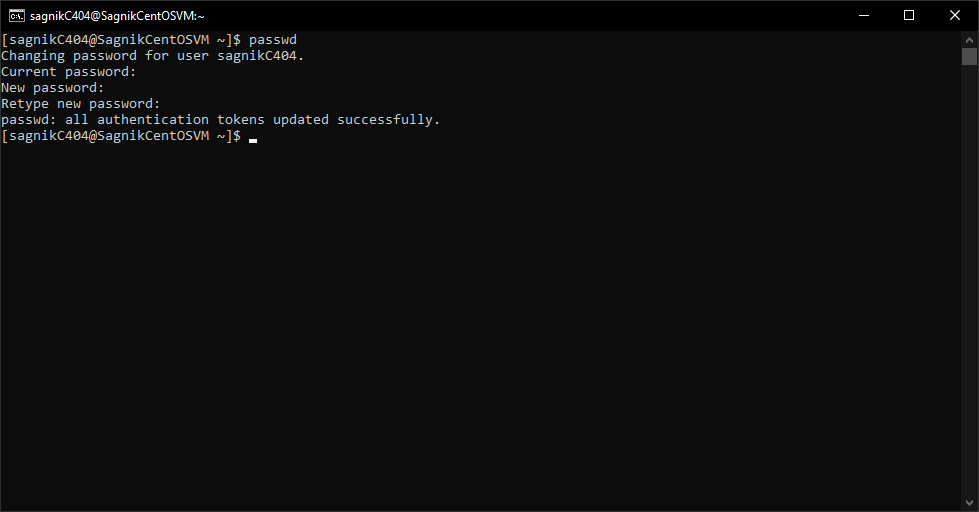
|  |
| --- |
| **Linux File Types** |

|  |  |  |
| --- | --- | --- |
| **Sl No** | **Name** | **File Symbol** |
| 1 | Regular File | - |
| 2 | Directory | d |
| 3 | Special File or Device File | c |
| 4 | Socket | s |
| 5 | Named Pipe | p |
| 6 | Blocked Device | b |
| 7 | Link | l |
| **Root** | | | | |

There are 3 things which are addressed as root on a linux system

|  |  |  |
| --- | --- | --- |
| **Sl No** | **Name** | **Explanation** |
| 1 | Root User (Account) | Root is the most powerful account on a Linux machine which has access to all the files and commands |
| 2 | Root Directory (/) | The very first and the top most directory in the Linux file system is also referred to as the Root |
| 3 | Root Home Directory (/root) | The root user account also has a directory named /root which is the home directory of the root user |

* **Changing Password Of Our Account**
* First we have to type the command passwd
* Next it will ask for the old password
* After successfully entering old password for the user account, it will prompt for new password
* After confirmation, the password will be successfully changed

****

|  |
| --- |
| **Absolute And Relative Paths** |

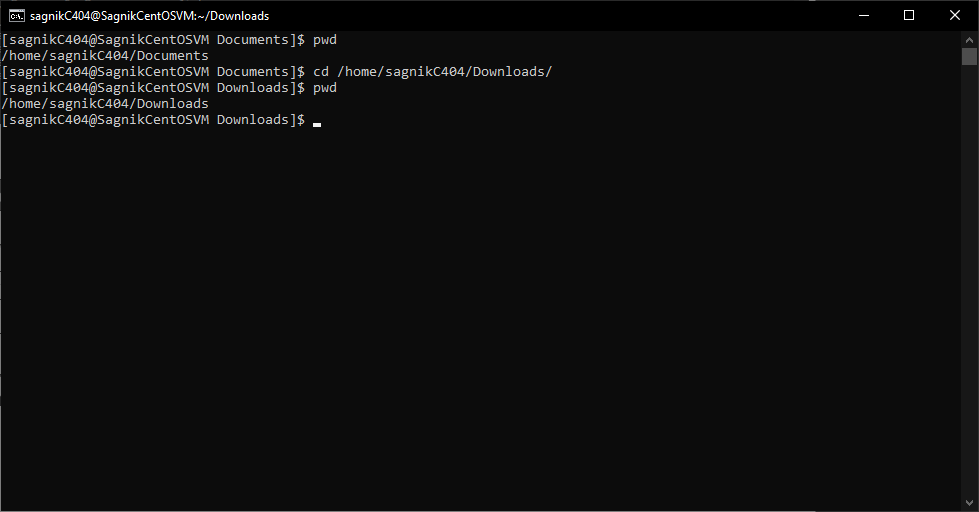
There are two ways to navigate a file system.

|  |  |
| --- | --- |
| Absolute Path | Absolute path starts from root. It has a ‘/’ in the beginning |
| Relative Path | Relative path to a file/directory depends on our present location. It is relative to our current position. It doesn’t start with a ‘/’ |

Let us suppose we are in the Documents Directory, and we want to move to the Downloads.

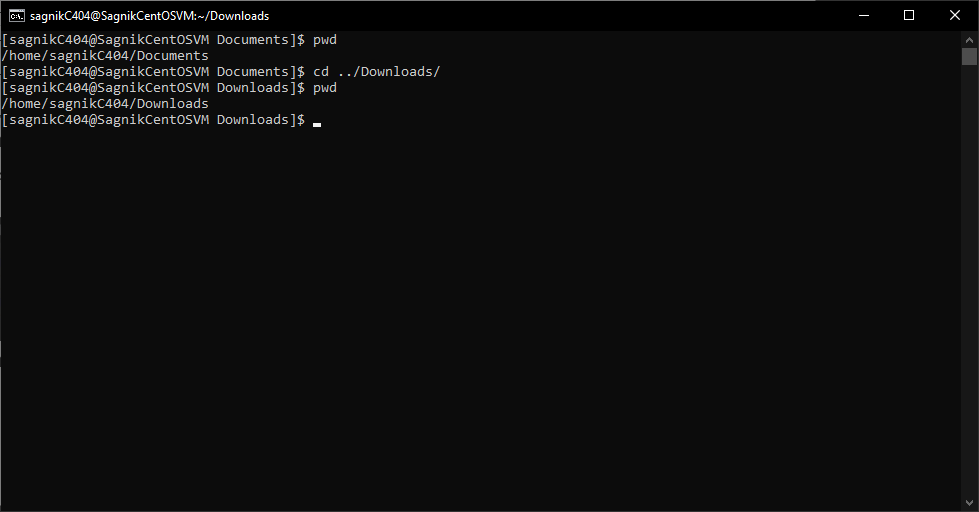
If we want to do it using the absolute path, we will do the following

cd /home/sagnikC404/Downloads/



If we want to do it using the relative path, we will do the following

cd ../Downloads/

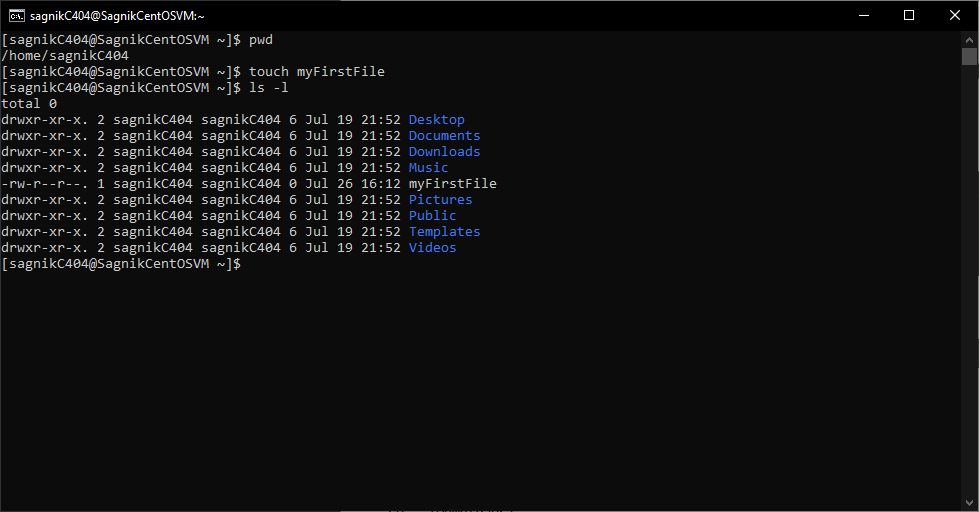


|  |
| --- |
| **Creating Files And Directories** |

* **Creating Files Using touch command**
* First we have to go to the location where we want to create a file.
* Next we need to write the following command

touch myFirstFile

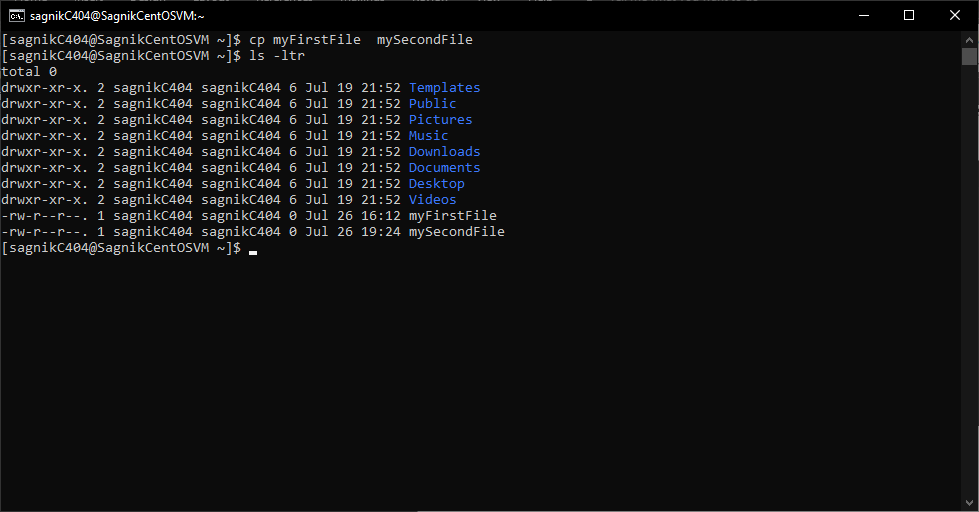
* Next if we list out the contents present in that directory, we will see our file named ‘my first file’ has been created successfully.



* **Creating Files Using cp command**
* First we have to go to the location where we want to create a file.
* Next we need to write the following command

cp mySecondFile myFirstFile

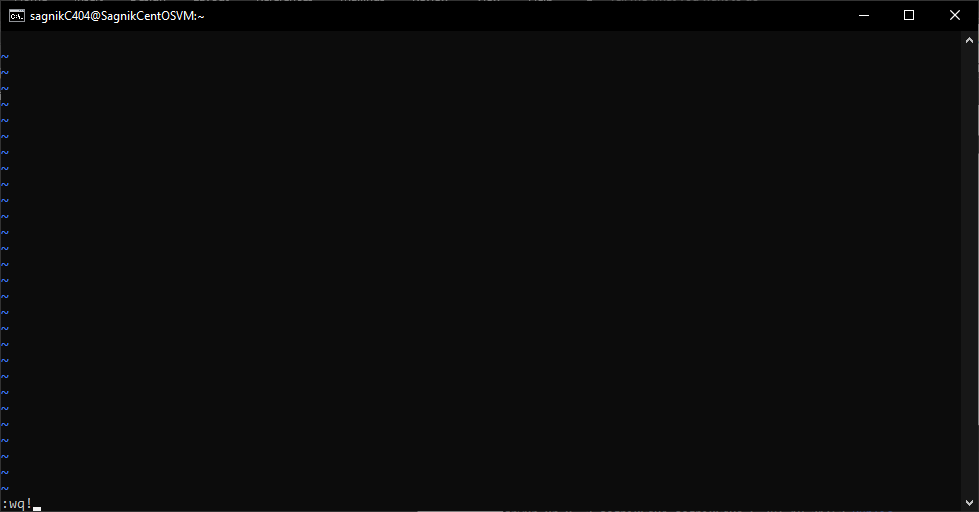
* It basically creates a copy of myFirstFile into another file named mySecondFile.

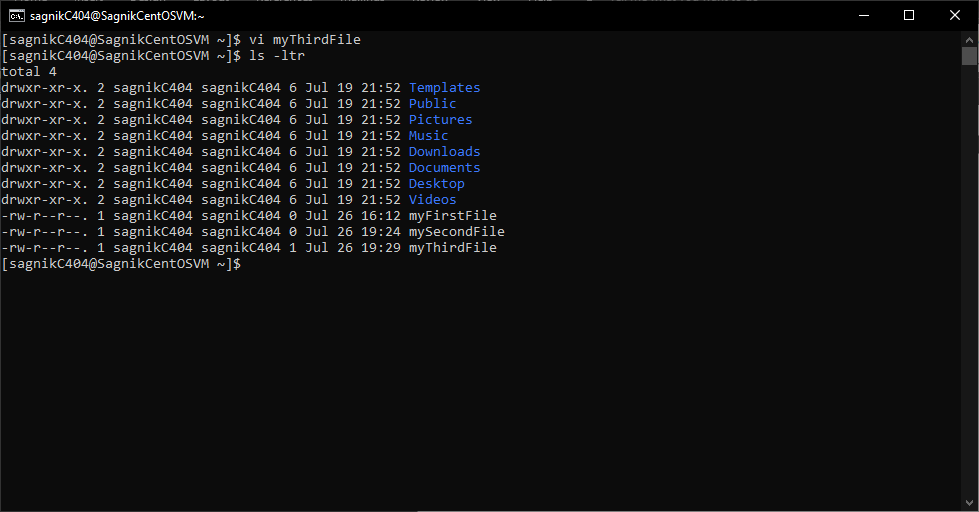


* **Creating Files Using vi command**
* First we have to go to the location where we want to create a file.
* Next we need to write the following command

vi myThirdFile

* It opens the file in vim editor.
* Now, to save the file, we can use :wq!

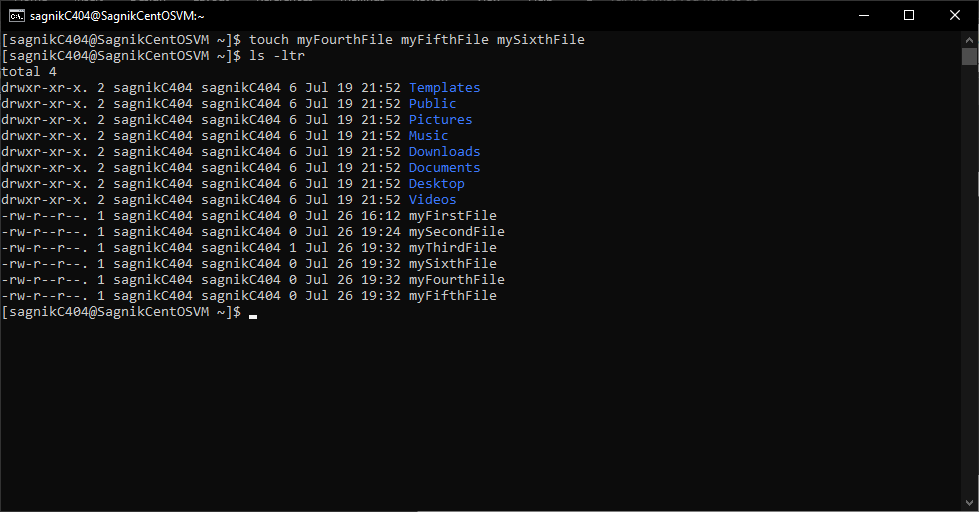




* **Creating Multiple Files Using touch command**
* First we have to go to the location where we want to create a file.
* Next we need to write the following command

touch myFourthFile myFifthFile mySixthFile

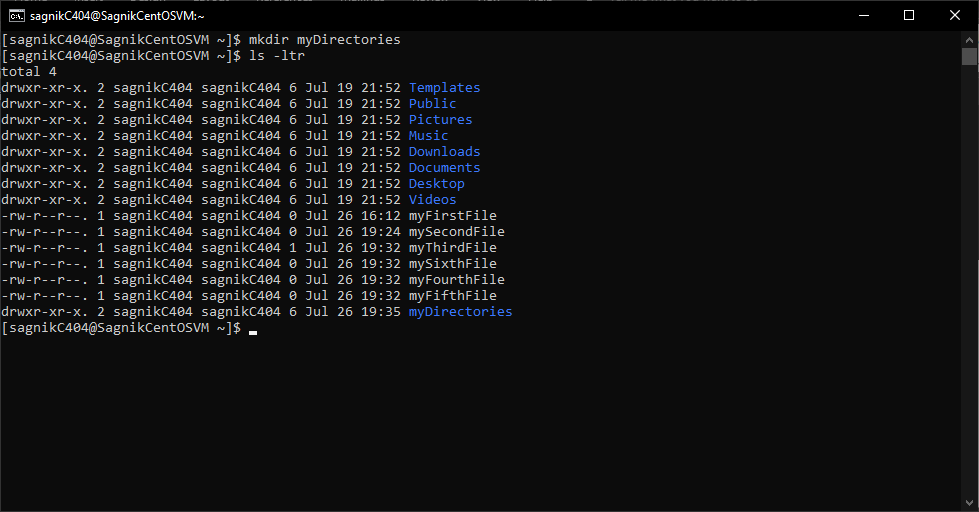
* Next if we list out the contents present in that directory, we will see our files have been created successfully.



* **Creating Directories Using mkdir command**
* First we have to go to the location where we want to create a file.
* Next we need to write the following command

mkdir myFirstDirectory

* Next if we list out the contents present in that directory, we will see our files have been created successfully.

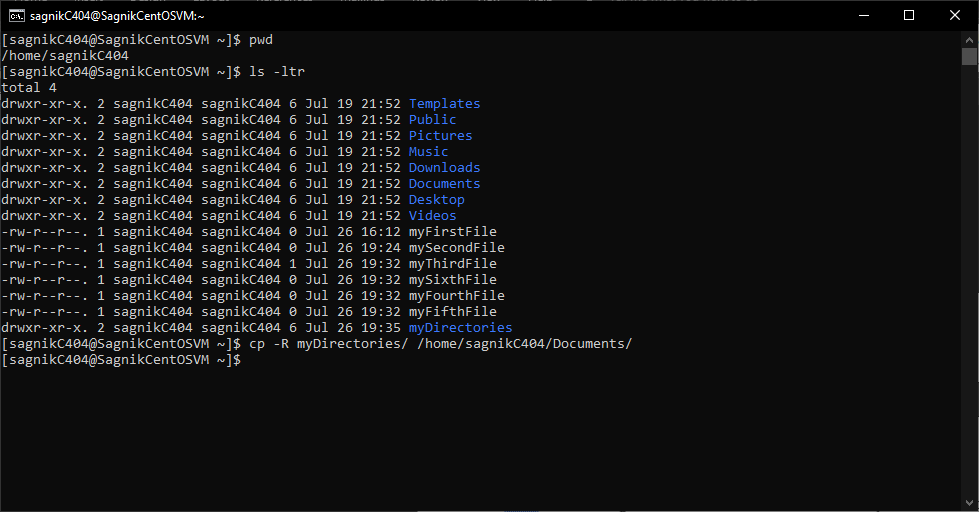


|  |
| --- |
| **Copying Directories** |

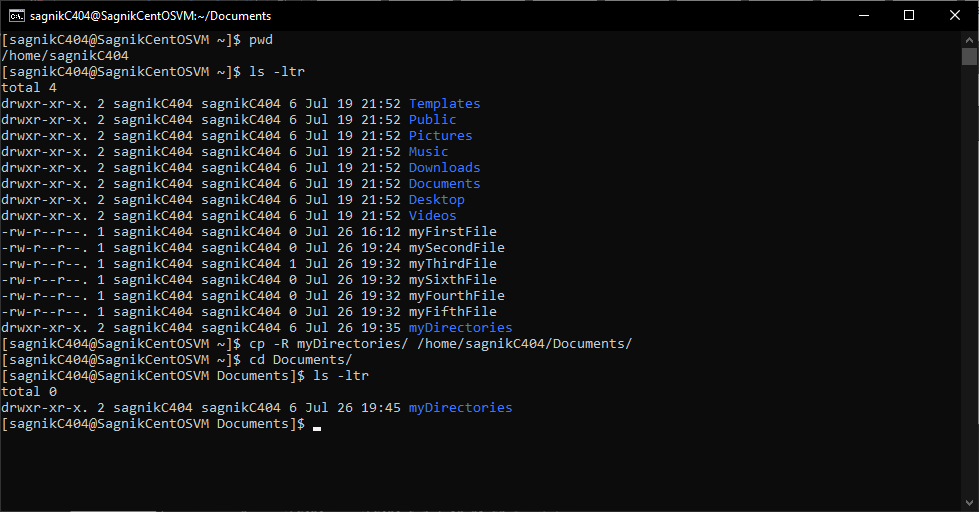
* **Copying Files Using touch command**
* First we have to go to the location where the directory we want to copy is present.
* Our directory is in our home folder. Let us move it to the documents folder
* Next we have to write the following command

cp -R myFirstDirectory /home/sagnikC404/Documents

* -R flag will ensure that everything inside the folder will be copied.
* Next we can go to the destination directory
* Next if we list out the contents present in that directory, we will see our file named ‘my first file’ has been created successfully.



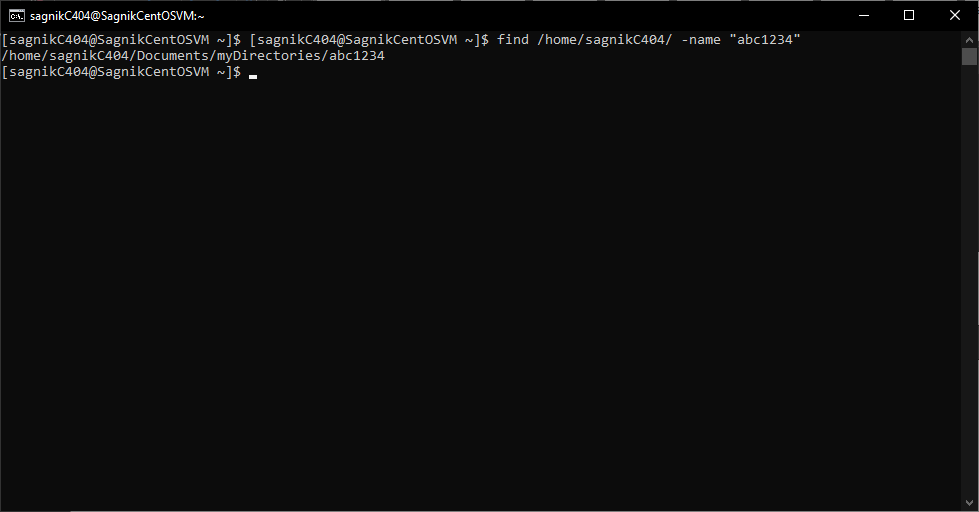
* Lets check in the destination directory



|  |
| --- |
| **Find Files & Directories** |

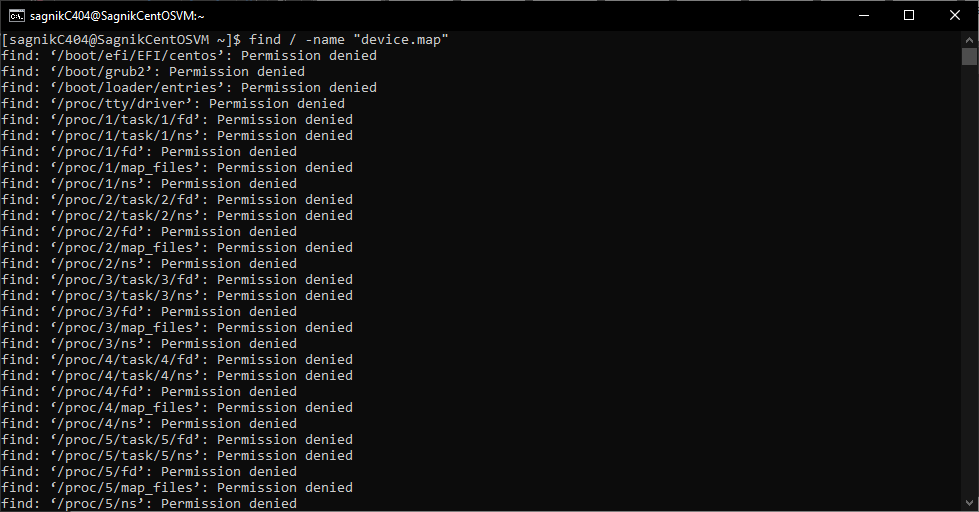
* **Find Command**
* Find command is used to find files and directories
* Let us suppose we have a file named abc1234 at /home/sagnikC404/Documents/myDirectories
* We want to find that file
* Next we need to use the following command

find /home/sagnikC404/ -name "abc1234"



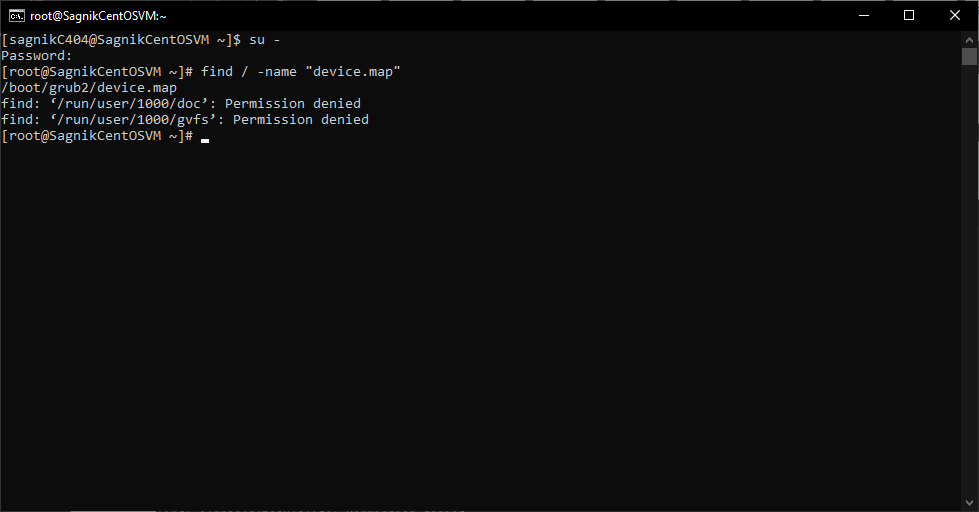
* Find command is has user restrictions
* If we need to find anything in a directory, where we don’t have the permissions, we will get permission denied
* For example let us try searching the file device.map

find / -name “device.map”



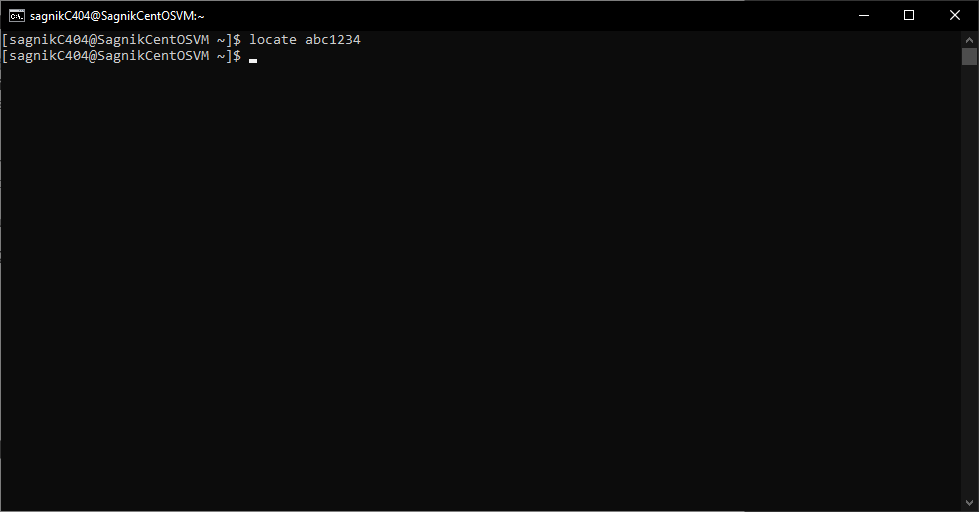
* To search things in those directories, we need to become root and then search
* To become root, we have to enter the following command su-
* Enter the password and on successful authentication we can get elevated to root user
* Now if we use the command, we will find those files

find / -name “device.map”



* **Locate Command**
* Locate command works in the same way as that of find command
* Except it maintains a database of the files and hence searching is faster in case of locate than that of find
* However the database needs to be updated else it wont be able to track them
* Let us suppose we have a file named abc1234 at /home/sagnikC404/Documents/myDirectories
* We want to find that file
* Next we need to use the following command

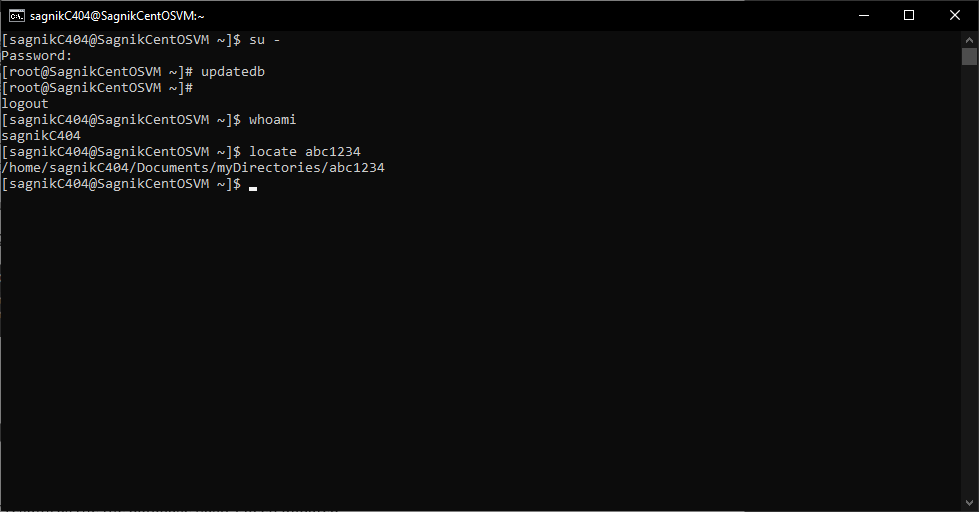
locate "abc1234"



* We don’t see any results as the file database hasn’t been updated
* We can update that using the following command

updatedb

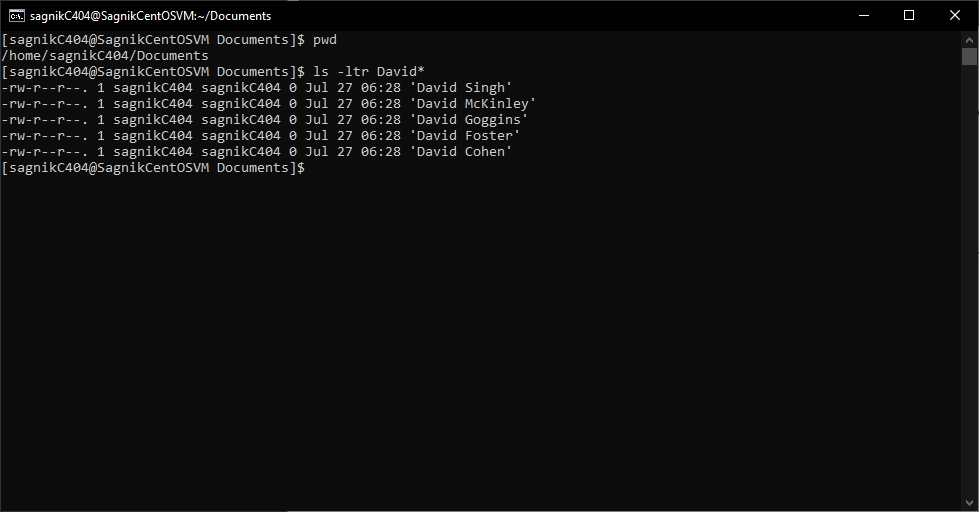
* We have to be the root user to do so
* Next if we run the locate command, it will find us the file



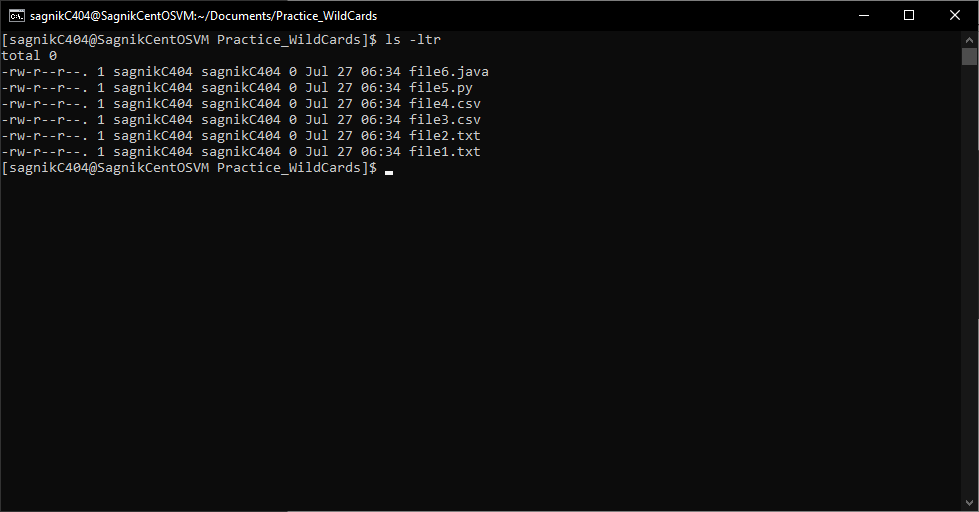
|  |
| --- |
| **Wild Cards** |

* **Wildcards**
* A wildcard is character that can be used as a substitute for any class of a character in search
* \* - represents 0 or more characters
* ? – represents a single character
* []- represents a range of characters
* **The Asterisk \***
* Let us suppose we have some files assigned to persons with same first name but different last names.
* To list them all, we can use the \* Wild Card.
* We have to write the command

ls -ltr David\*

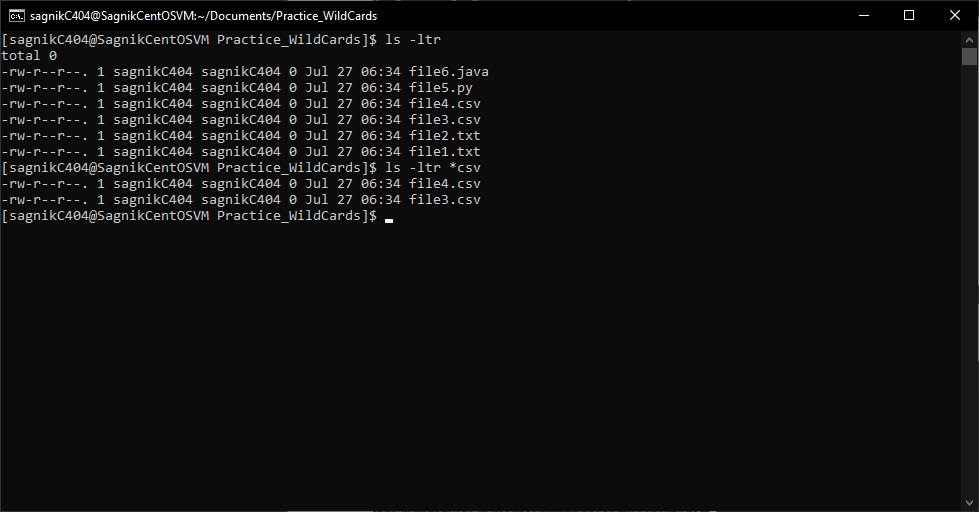


* Let us suppose we have some files with different extensions like csv, java, py, txt.

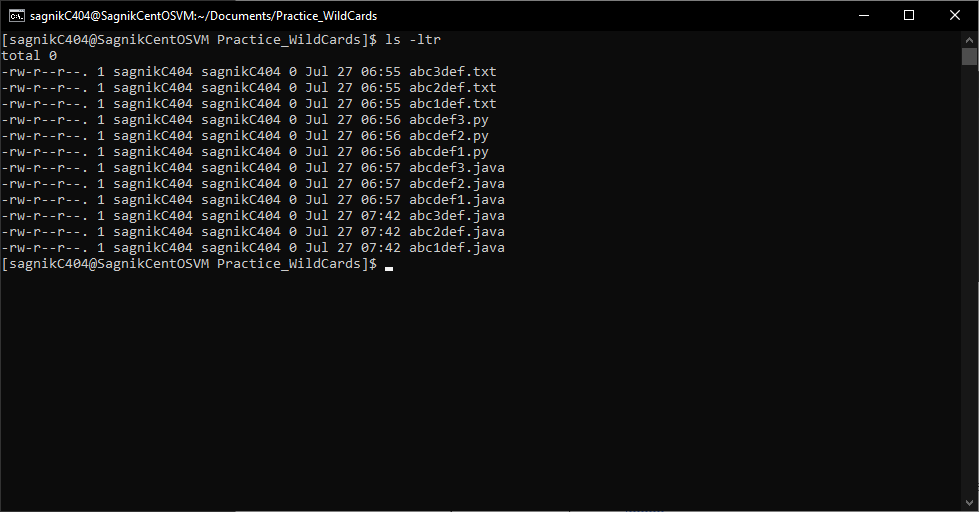


* Out of all of them, we want to list out the csv files only.
* We have to write the command

ls -ltr \*csv

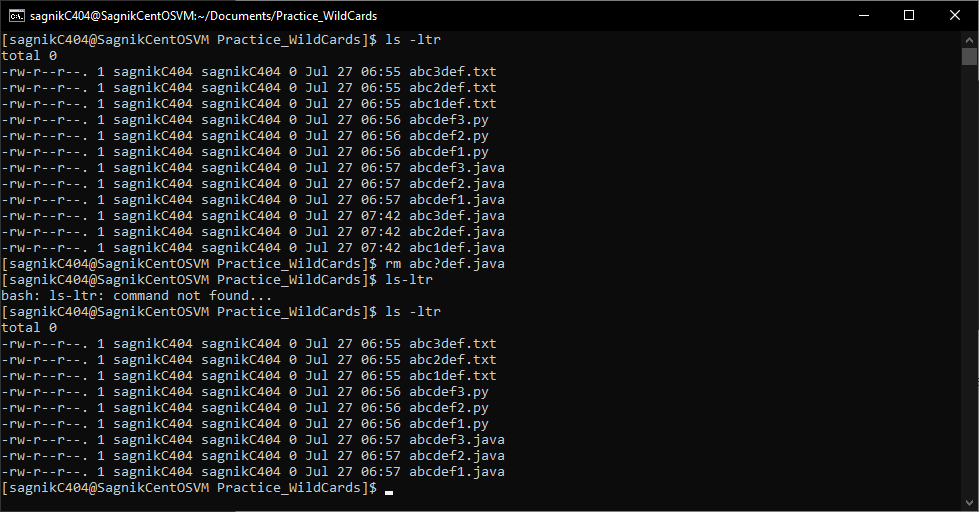


* **The Question Mark ?**



* Let us consider the following list of files.
* We want to delete those files which has a number between abc and def followed by a .java extension
* To do this kind of precise selection, we use the ? wildcard operator
* We have to write the command

rm abc?def.java

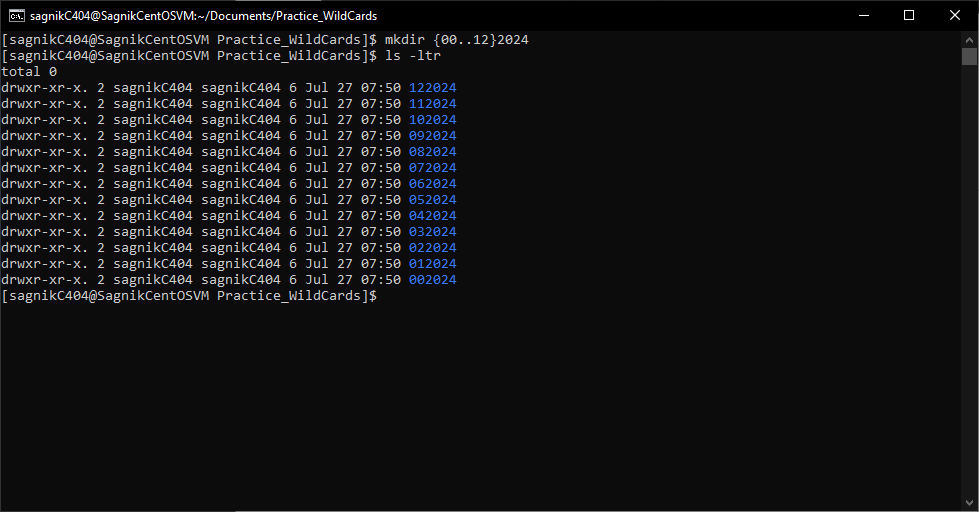


* We can see from the list that the specified files starting with abc followed by a number and then def and finally having a extension .java are removed.

* **The Range { }**
* Let us suppose we have to create directories for every month from January to December 2024.
* The naming convention is as follows 012024 022024 032024 and so on
* To create them all, we can use the { } Wild Card.
* We have to write the command

mkdir {00..12}2024

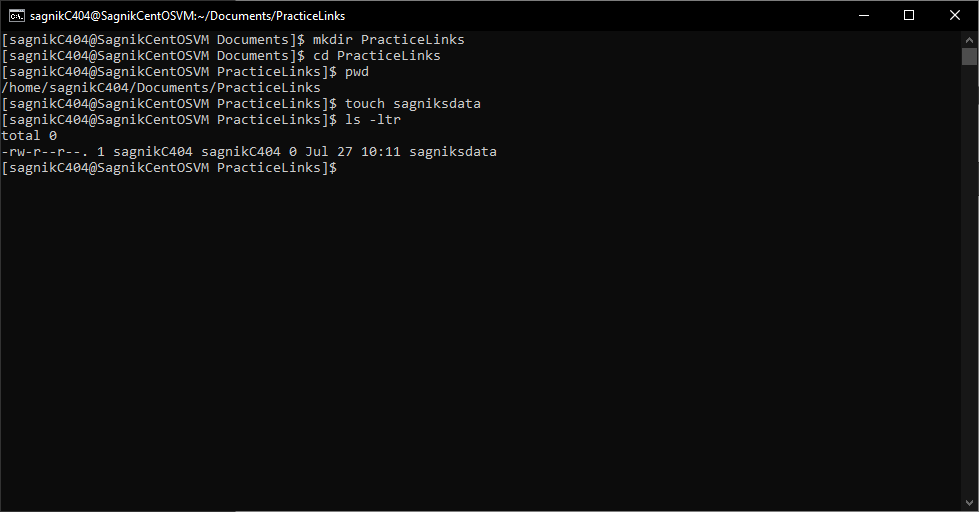
* If we list out the contents , we will see the following



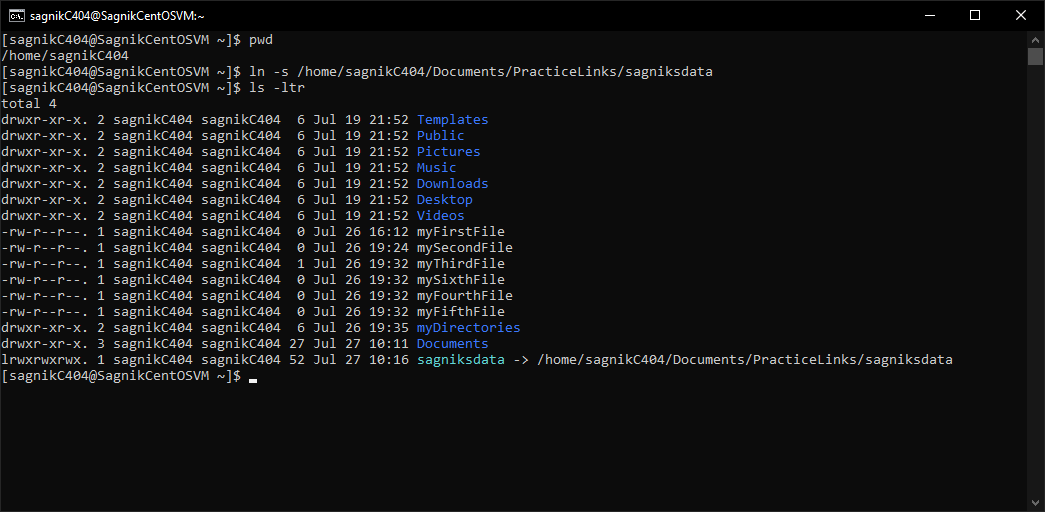
|  |
| --- |
| **Soft Links and Hard Links** |

* **iNodes**
* iNode is a number or a pointer on a file on the hard disk
* **Soft Link**
* A soft link is a link to a file which is removed if the file is renamed , moved or deleted
* A soft link is created using the command ln -s

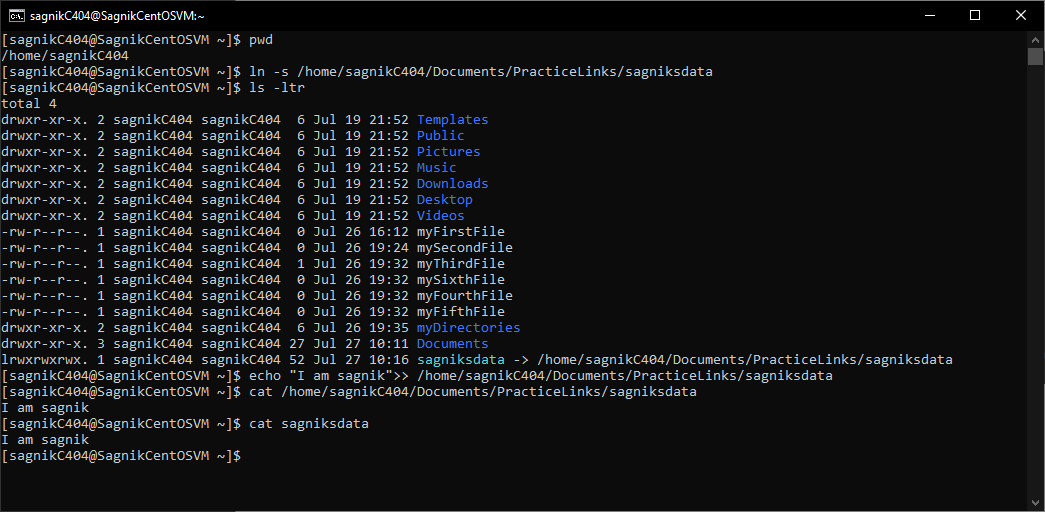
Let us suppose we have a file named ‘sagniksdata’ at the given location.



Let us go to the home directory and create a soft link to that file

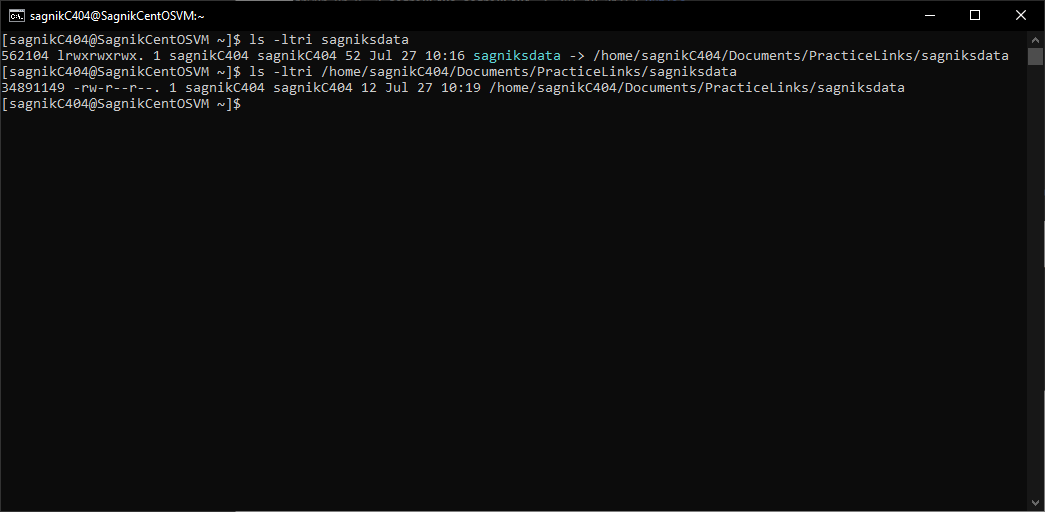


Let us make some changes to the original file by putting in some content

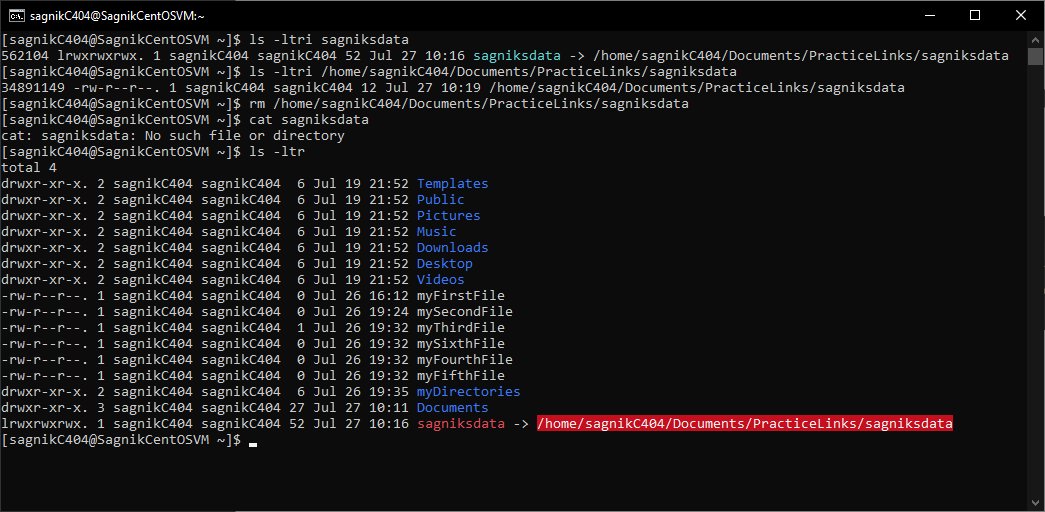


As we can see, changes appear in the link as well

Now if we try to retrieve the iNode number of the original file and the link, we will see that iNode numbers assigned to them are different.

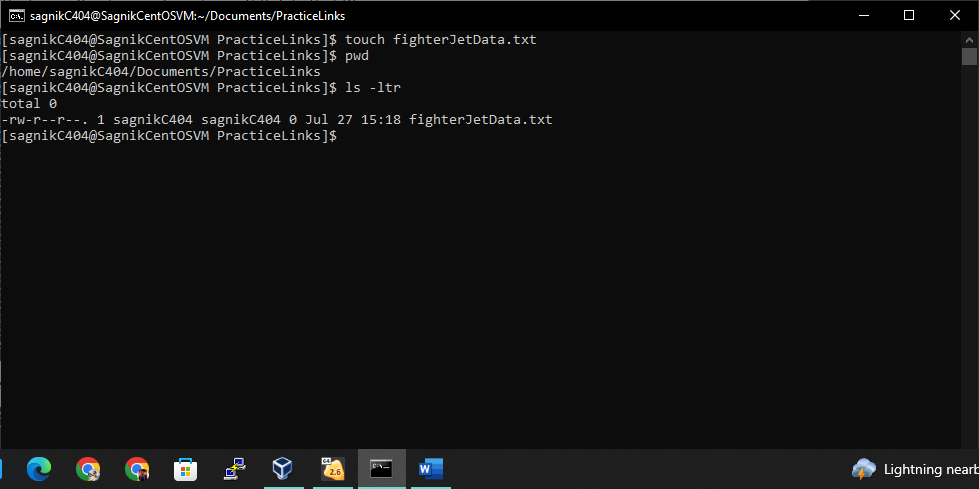


Hence, if we delete the original file, the link will start to malfunction

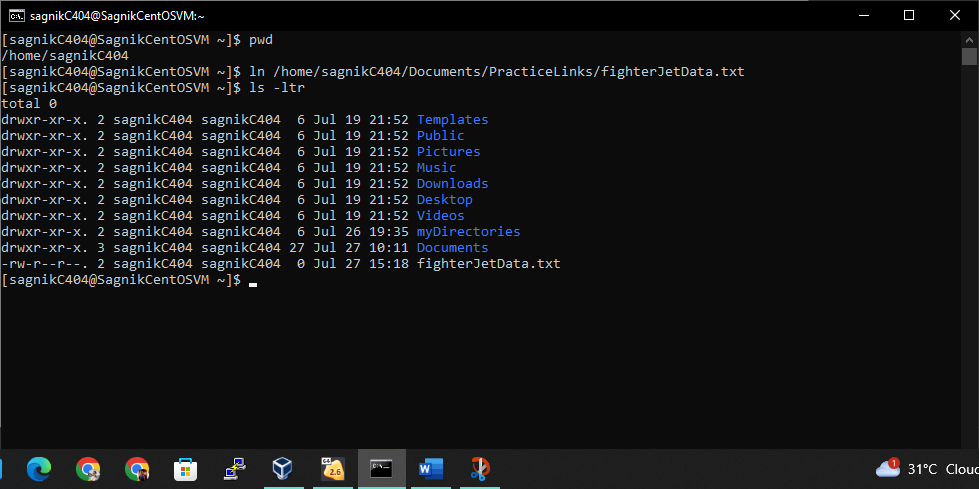


* **Hard Link**
* A hard link is a link to a file which is unaffected if the file is renamed , moved or deleted
* A hard link is created using the command ln

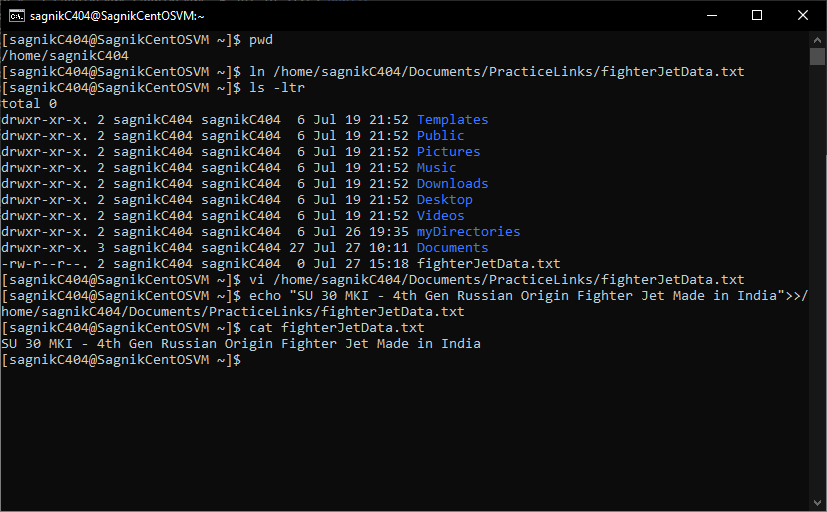
Let us suppose we have a file named ‘fighterJetData’ at the given location.



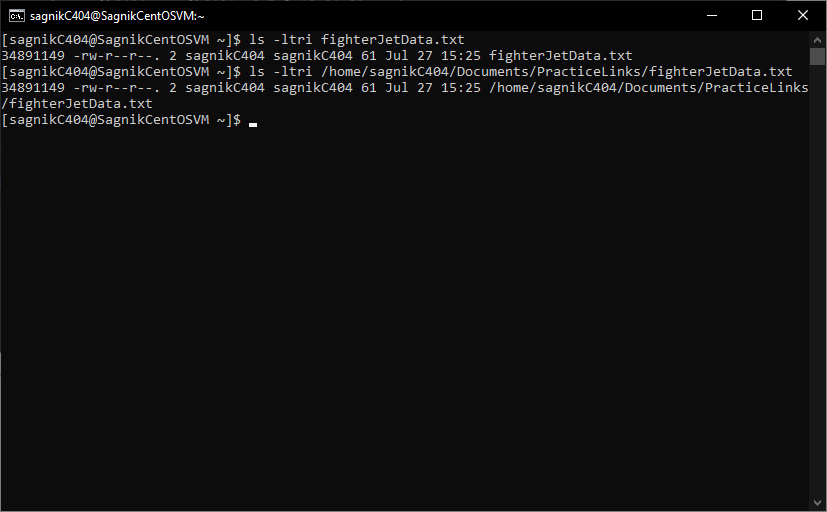
Let us go to the home directory and create a hard link to that file



Let us put some content in the original file and if we check the contents of the link, we will see the same thing.



If we check the inode values of the original file and the link, we will see



The iNode values are the same for both the files. Hence if we delete the original file, the link will keep functioning normally

