

Linux Workshop

Written by Naavin

July 24, 2022



About Me

Just a quick introduction.

- ▶ My name is Naavin.
- ▶ Final year Bachelor of Computer Science Student at Monash.
- ▶ Self-proclaimed Linux enthusiast. ^{I use Arch btw.}
- ▶ Besides using Linux in various units in Monash¹, I've been using a Linux distribution as my main OS for years, long before joining Monash.

¹I didn't take the data science units though where I've been told they used Linux sometimes.

What this will cover.

and not cover.

I will cover

- ▶ Brief history of Linux
- ▶ Brief overview on how to install it.
- ▶ How to use the Linux Terminal
- ▶ Several Linux terminal commands.

I will *not* cover

- ▶ Every single possible thing that you will need to know to do all of the Monash units. (Hopefully, you will be soon be comfortable enough that it is easy to learn yourself though.)

What is Linux

- ▶ Linux is kernel, not an operating system.
- ▶ There are many different Linux distributions (which are like operating systems), with slight differences (init system, Desktop environment, package manager, etc.)
 - ▶ Debian
 - ▶ Arch Linux
 - ▶ Fedora
 - ▶ openSUSE Tumbleweed/Leap
- ▶ Not binary cross-compatible with other operating systems like Mac OS or Windows, i.e: You can't run a ".exe" program.²
- ▶ Open-Source
- ▶ Many devices run on Linux, for example, Many Web Servers and Android uses the Linux kernel. The recent "Steam Deck" also uses Linux (a variant of the Arch Linux distribution).
- ▶ You might need to use it because some programs are mainly written with Linux in mind.

²There is a program called "wine" that will allow you to run Windows programs.

What is Linux

Linux is a Unix-based³ kernel written by this guy, Linus⁴ Torvalds, first released in 1991. He still works on the Linux kernel to this day, though there are *many* more contributors now.



Figure: A picture of Linus Torvalds.

Fun fact: He was the same person who invented “git” to aid with development of Linux, because of limitations with other version-control systems at the time.

³MacOS is also Unix-based based, which is why you might see some similarities.

What is Linux

Linux was written by this guy, Linus⁵ Torvalds, first released in 1991. He still works on the Linux kernel to this day, though there are *many* more contributors now.

⁵Yes, with an “s” not an “x”

Stallman

Another key figure is Richard Stallman, a free software movement activist who launched the GNU Project, which included things like the GNU Compiler Collection (gcc), GNU Debugger (gdb), GNU Emacs (text editor) GNU shell utilities, etc.⁶



Figure: A picture of Richard Stallman.

⁶Linus disagrees. The guy is a bit eccentric and extreme in his beliefs on free software, but he is really smart and did well in the famously hard Math 55 course in MIT.

GNU+Linux?

He claims that his software is intertwined with Linux, and calling it just Linux is disingenuous, instead calling it “GNU+Linux”.⁷

I'd just like to interject for a moment. What you're referring to as Linux, is in fact, GNU/Linux, or as I've recently taken to calling it, GNU plus Linux. Linux is not an operating system *unto itself*, but rather *another free component of a fully functioning GNU system made useful by the GNU corelibs, shell utilities and vital system components comprising a full OS as defined by POSIX*. Many computer users run a modified version of the GNU system every day, without realizing it. Through a peculiar turn of events, the version of GNU which is widely used today is often called “Linux,” and many of its users are not aware that it is basically the GNU system, developed by the GNU Project. There really is a Linux, and these people are using it, but it is just a part of the system they use.

Linux is the kernel: the program in the system that allocates the machine's resources to the other programs that you run. The kernel is an essential part of an operating system, but useless by itself; it can only function in the context of a complete operating system. Linux is normally used in combination with the GNU operating system: the whole system is basically GNU with Linux added, or GNU/Linux. All the so-called “Linux” distributions are really distributions of GNU/Linux.

Figure: His famous rant some snarky person will say whenever you say “Linux” on the internet. There is also a less-famous rebuttal rant to this.

⁷Most Linux distros, with rare exceptions like Alpine Linux, have the GNU “toolset” preinstalled since it is so important.

What distro?

Finally to installation, but first you need to choose a distribution.



Figure: Logos of distributions. I've seen many "collages of logos" online erroneously including FreeBSD as well, which isn't a Linux distribution..

What distro?

If you're using it only for Monash units, I *strongly* recommend you to just install whatever they recommend or use the OS VM they provide. This way, TAs can easily reproduce any problems you might face and ask for help with. I will talk more about this later on.

If you want to install it for yourself, read on!

What distro?

If you're doing it for your own benefit go ahead and factor these things out.

- ▶ **Community** - How active is the community so that you can probably find a solution to problems you face with the distro? Popular distros like Linux Mint and Ubuntu have more community support than very niche distros that less people use.
- ▶ **Ease of Use** - Certain distros like Arch Linux or Gentoo were designed for advanced users, and may be harder for a newcomer to learn.
- ▶ **Purpose** - Some distros are tailored to specific use cases. If you're doing this for privacy reasons⁸, Qubes OS or Tails may suit you. If you're a security researcher, Kali Linux comes preinstalled with many security-related tools. If you're a data scientist or a gamer, Pop OS! is designed to be easy to fully support AMD and Nvidia⁹ GPUs.

⁸Avoiding sending telemetry to Microsoft on Windows or giving your data to Apple

⁹Nvidia linux support is notoriously spotty. Thanks Nvidia.

How to Install.

There are various ways you can use Linux.

- ▶ Virtual Machine software (Virtualbox / VMWare / qemu)
- ▶ Dual-booting / Replacing your OS on your machine.
- ▶ Remotely connecting to a Linux machine (Usually Monash doesn't provide this though except for special circumstances)
- ▶ WSL2 (via Microsoft Store) on Windows / Cygwin / MingW / whatever.

Just for demo purposes, I've also found a WebAssembly virtualisation for some Linux distributions here:

<https://github.com/copy/v86>. It only provides a terminal interface, but you can use it to play along later on when we talk about Terminal commands, if you wish.

Installation

Virtual machine Software

The typical steps include this:-

1. Ensure Hyper-V is enabled on your machine (search online on how to enable it, you *might* need to go to your BIOS settings to enable it.)
2. Download an “.iso” file from the distro of your choice.
3. Create a new entry in your VM Software.
4. Create a “virtual” hard drive for that entry.
5. “Mount” the iso file as a bootable CD/USB drive.
6. Start the VM, it should bring you into the “Live CD” environment.
7. Depending on the distro, there will be a “demo” where you can use the OS to see if you like it (but doesn't persist changes after a reboot), and also a way to “install” the distro onto your hard drive, at which point you follow the instructions.¹⁰

¹⁰If they ask for Swap Space, this is additional emulated “RAM” in the filesystem if all your real RAM is used up. It is controversial how much to use, if any at all, but keep in mind that Hibernation is only possible if $Swap \geq RAM$.

Installation

Virtual machine Software

If you think that's hard, don't worry. Monash usually provides pre-installed virtual drives you can directly import (e.g: vdi files for Virtualbox), so it is way less stuff you need to do.

1. Install the virtual hard drive they provided.
2. Import it into Virtualbox/Whatever.
3. Launch the Virtual Machine.

One of my units pointed us here to get pre-made images here:

<https://www.osboxes.org/virtualbox-images/>

Of course, you place your trust into them for not adding anything malicious into the image, since they are a 3rd party.

Installation

Virtual machine Software

Notes:

- ▶ If it doesn't work, you may need to play around with the Settings in Virtualbox/Whatever you use. You can probably find help on PDF instructions given to you, or on edstem forums; or you can ask your TA during your session or email.
- ▶ If you use a Mac with an Apple M1 or Apple M2 chip (ARM-based), you'll run into problems since the virtual machines Monash provides usually are based for x86 CPU architectures. You can install an ARM-based version of the distro if it exists, but I wouldn't risk it if the units you're doing use certain software that might have subtle differences in non-x86 CPUs. I'd suggest using a software called UTM¹¹ which should emulate x86, or borrow/use an old x86 laptop from someone, or ask your TA for advice.

¹¹It uses qemu under the hood.

Virtualbox

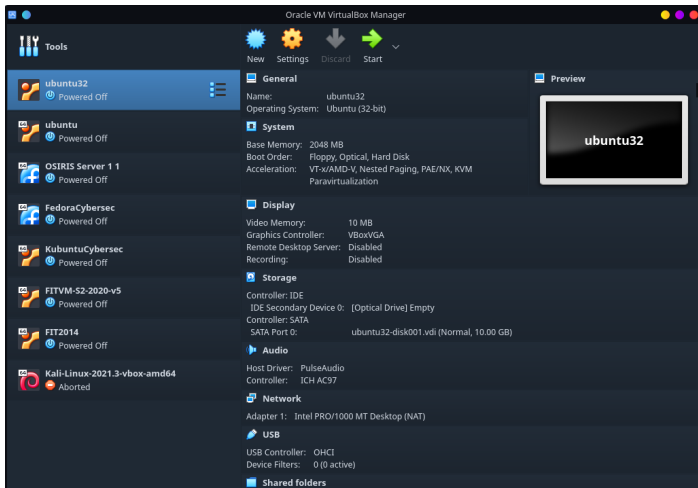



Figure: Note: You might need to play around with the “Settings” and mess around with the allocated RAM or display settings.

Installation

Dual-booting

Pretty similar to installing it yourself on a Virtual Machine. I strongly advise you to back up all your files first prior to doing this in case you mess up.

- ▶ If you plan to dual-boot (e.g: be able to use either Linux or Windows), you need to “shrink” your Windows partition to make room for Linux.
- ▶ Download the iso file.
- ▶ “Burn” the iso file into a USB drive (or CD) ¹² to make a bootable USB drive using software like Rufus or balenaEtcher, and plug it in to your machine.

¹²Note that burning will lose all files on your USB drive / CD! 

Installation

Dual-booting

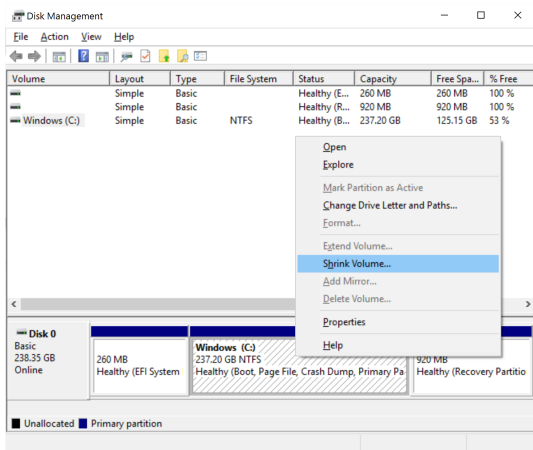


Figure: I suggest backing all your files first.

Installation

Dual-booting

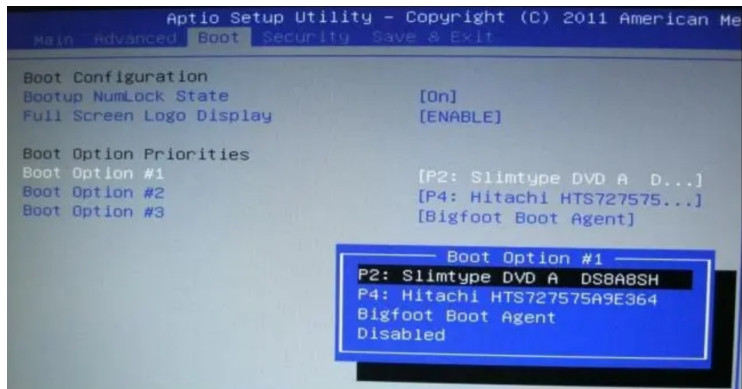


Figure: Change your boot order.

Installation

Dual-booting

- ▶ Enter your BIOS settings / UEFI Firmware by mashing a key combo specific to your machine on boot (usually F12 or similar). and look for a “boot” tab and move the boot order/priority of your USB drive to have higher priority. You might also need to disable secure boot.
- ▶ Restart the machine, and it should boot into the USB, at which point instructions are similar to VMs.
- ▶ Note: For Macs, I heard there is something called “Boot Camp” that helps with dual-booting for intel based macs.

Virtualbox

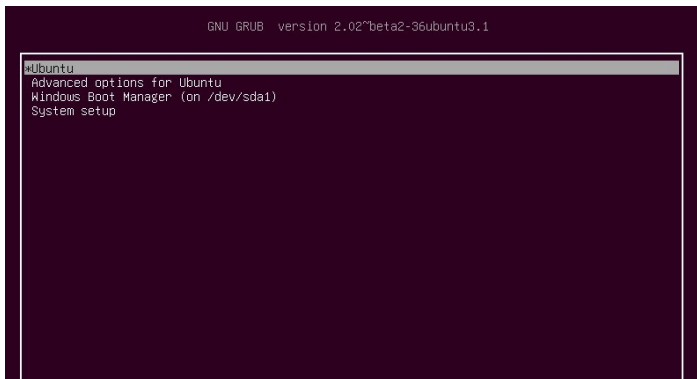


Figure: When you start you see the option to choose what OS to boot to.
Note: There are many themes you can use besides this stock one.

Installation

Remote-Connection

I don't have any experience with how Monash does this, but presumably there will be a web page (like MoVE) or software to install so that you can remotely connect to their Linux machines. I am sure there are commercially available “cloud” solutions as well.

If you only want a Linux Terminal without the GUI, there are many services like DigitalOcean that provide a container for a Linux distribution that you can remote into for a slight cost, though these were designed for things like being web servers or services.

Installation

Microsoft Store

The Microsoft Store on Windows provides something called “WSL2” for various distros, which is kind of like a VM of Linux distros. You can just search for “Ubuntu WSL” and install it, and you should have a new program called “WSL2” that gives you a Linux Terminal. ¹³

As of time of writing, they only provide the terminal Linux interface in stable public releases. The nice thing about WSL2 is that it is more lightweight than a VM and with better performance, and it uses the same filesystem as your native machine (Windows).

¹³Again, you might need to enable Hyper-V.

Linux Terminal

For the remainder of the session I will talk about the Linux Terminal. If you're on a VM with a graphical environment, you'll need to use the terminal via a *terminal emulator*, which is usually preinstalled. Some common ones include:-

- ▶ Gnome Terminal
- ▶ Konsole
- ▶ XTerm
- ▶ Alacritty

Just look for a search bar and search “Terminal”, or look for a terminal icon somewhere, and it will probably work fine.

Linux Terminal

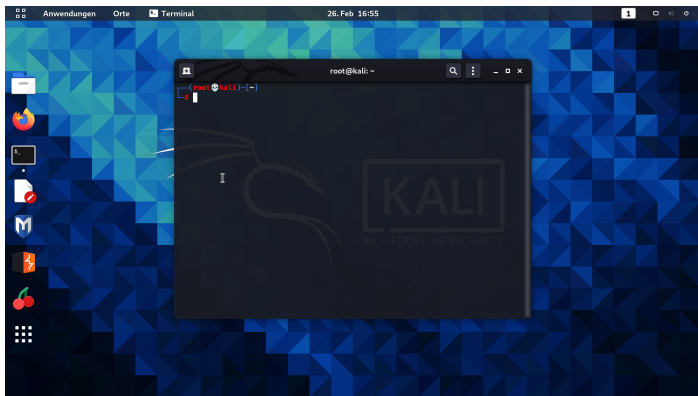


Figure: Example terminal.

Depending on your distribution, it might look different (e.g: not translucent, it might be white instead of black, the text on it might be different, etc.)

Filesystem

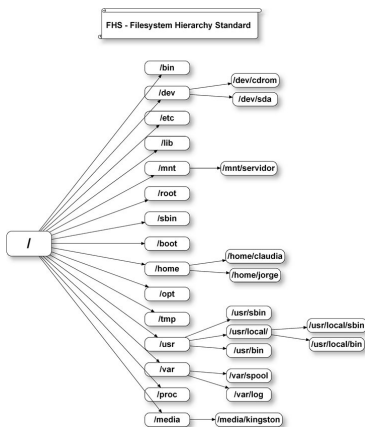


Figure: Example terminal.

`"/home/username"` is where you spent most of your time. `"/bin"` is where most of your programs live.

<https://refspecs.linuxfoundation.org/fhs.shtml>

Filesystem Navigation

- ▶ When you start a terminal, it usually automatically places you in your “/home/username/” directory.
- ▶ If you ever are asked to “run the command in this folder”, you need to make sure your terminal is in the right folder.
- ▶ You can type “pwd” and hit enter to see the directory you are in. (print working directory).

```
$ pwd  
/home/naavin
```

Filesystem Navigation

- ▶ “pwd” was actually a program that you just ran, located at “/bin/pwd”.
- ▶ Other commands include:-
 - ▶ **ls** - lists files/folders in your current directory.
 - ▶ **whoami** - prints out the username of the current user.
 - ▶ **tree** - lists files/folders in your current directory and its files/directories recursively.
- ▶ Most commands also accept more inputs (options) after it.
 - ▶ **ls** <**directory**> - lists files/folders in <directory>.
 - ▶ **man** <**program**> - shows you a manual page for the command.
 - ▶ **cd** <**directory**> - changes your current directory to <directory>. Note you only need to state the path relative to your current path.
 - ▶ **touch** <**filename**> - creates an empty file <filename>.
 - ▶ **echo** <**text**> - prints out <text>.
- ▶ Most programs also accept a `--help` option to show all available options.

Filesystem Navigation

- ▶ I said for “cd” (and most programs taking in a directory), you only need a *relative* path. This means, assuming you’re in the “/home/naavin” directory, the following are all the same.

```
$ cd documents/homework
```

```
$ cd ./documents/homework
```

```
$ cd /home/naavin/documents/homework
```

```
$ cd /home/naavin/documents/./documents/homework
```

- ▶ “./” refers to the current directory, and “../” is referring to the parent’s working directory.

More commands

- ▶ **rmdir** <directory> - removes/deletes the directory <directory> (if it's not empty).
- ▶ **rm** <filename> - removes/deletes a file.
- ▶ **mkdir** <directory> - creates the <directory> .

Note: mkdir will only make the “last” folder in the input you give it. If it specifies a non-existing place to create the folder, it won't work unless you use the option “-p”.

Exercise

1. Create a directory¹⁴ called “good”
2. Create a directory called “bad”
3. In the “good” folder, create a folder called “books”
4. In the “good” folder, create a folder called “movies”
5. Do the same in the “bad” folder.
6. Create an empty file “cursedchild.txt” in the “bad/books” category.
7. Create an empty file “titanic.txt” in the “good/movies” category.
8. Create an empty file “morbius.txt” in the “bad/movies” category.
9. Remove the file “morbius.txt”.
10. Remove the directory “bad/movies”.

¹⁴term used interchangeably with “folder”

More on “options”

1. Not limited to one “option” / argument. You can have many!
2. `ls -l -a` has two arguments, “-l” and “-a”.¹⁵
3. Note that if you want to have a single argument with a space in it, you need to wrap it in quotes, e.g.
`echo "Single Argument"`.
4. There are some special characters like `;` `*` `|` `&` `\`¹⁶ that you cannot use without escaping it or using the “quotes” trick mentioned.

¹⁵btw they made the program such that you can combine it to just “la”

¹⁶non-exhaustive

More on “options”

1. **cp** < **sourcefile** > < **destfile** > - copies a file.
2. **cp -r** < **sourcefile** > < **destfile** > - copies recursively (a directory for example).
3. **mv** < **source** > < **dest** > - moves a file/folder.

What is this # and \$?

Occasionally, when reading online resources or instructions from your tutor, they might ask you to type in something like this:-

```
$ echo "Hello!"
```

Or something similar to this.

```
/home/user/some_folder $ echo "Hello!"
```

Or something similar to this.

```
~/some_folder $ echo "Hello!"
```

What they want is to only type in the “echo ” Hello World”” part, while also being in the some_folder directory! ¹⁷

¹⁷If you're wondering about the tilde, it's a short-form for your home directory.

What is this # and \$?

But sometimes you might see this hash symbol.

```
# apt install python
```

What this means is that they want you to be running a *root* when running this command. ¹⁸.

You can run “sudo su” prior to executing that command to get a “root shell”¹⁹, or prepend the command with “sudo”, which will prompt you for your password. Note that doing this is dangerous, be careful.

¹⁸This is similar as “Running as Administrator” on Windows

¹⁹Running whoami will say your user is root now.

rm rf

This will give you an error.

```
$ rm -rf /
```

If you read the manual,

- ▶ “-r” stands for recursive.
- ▶ “-f” stands for “force”.

So in theory this will delete every file on your system. However, if you run this as a regular user, you won't have sufficient permissions to delete most of the files/folders. So it won't work.

rm rf

This however, might work if you're using an older linux distribution.

```
$ sudo rm -rf /
```

- ▶ Now you have full permission.
- ▶ Eventually they safeguarded this by explicitly checking for this and not allowing deleting everything, *unless* you specify a special option.

```
$ sudo rm --no-preserve-root -rf /
```

Running a program

Say you have a program in your current directory that you want to run.


```
$ ls
. ../ someprogram
$ program
Command not found!
$ ./program
Program running
```

- ▶ The reason programs like “pwd” don’t need a “./” before it is because they’re in your PATH environment variable (explained in later slides).
- ▶ You may need to run “chmod +x program” first to make the file “executable”.

Package Managers

In Windows-land, there is an asinine culture of downloading all your software from various EXEs from the Internet if you need a program.²⁰ In Linux-land, you generally use a package manager to install new software, with benefits like being able to update all your software at once, and not accidentally downloading a virus masquerading as another software on the Internet.

I mentioned before that there are slight differences in distributions, and this is one of them, as different distributions usually use different programs for package management, and also provide different “app stores”. Debian and its derivatives (Ubuntu, Linux Mint, Kubuntu, ...) use “apt”, which you may occasionally see the old name “apt-get” online.

²⁰Somewhat recently, Microsoft introduced Winget to solve this, and there have been third-party solutions like Chocolatey for a while. 

Package Managers

Package managers

There are several. Don't pull an LTT and wonder why "apt" doesn't work on Arch-based distros.

- ▶ "apt" for debian-based distros (Ubuntu/Mint/Kubuntu, etc.)
- ▶ "pacman" for arch-based distros.
- ▶ "dnf"
- ▶ "yum"

Debian-based distros are the most common, pay attention. Note that you can also install an executable manually (ELF file) or build it from source and run it, and it might work if it has executable bit set, and the file is compatible.

There are also other ways like AppImages and flatpaks or the snap store, which I won't talk about.

apt Usage


You might sometimes see “apt-get”, unless you’re running on an older distro, you can just replace it with “apt”.

```
# apt install python  
# apt purge python
```

Note you need to be root to install things.

Exercises

1. Install a package called “neofetch” ²¹
2. Run neofetch and observe the result.
3. Look into neofetch’s documentation (Hint: “neofetch –help” or “man neofetch”)
4. Figure out how to change what distro ascii art to print.
5. Run the program again, but change the distro ascii art to Ubuntu.

²¹The v86 link I sent isn’t able to use the internet unfortunately. 

Environment variables

- ▶ Some programs may use environment variables instead.
- ▶ `$ KEY=VAL ./program`
- ▶ You can keep the environment for the session with `export KEY=VAL`

grep Usage

Someone suggested giving a brief overview of this. You can also use ripgrep or silver surfer or other similar tools.

```
$ neofetch --help
```

```
$ neofetch --help | grep "ascii"
```

The output of “neofetch –help” is *piped* into “grep” and it searches for lines with the text “ascii”.²²

²²There is also ; for executing command sequentially and & that will only execute the latter command if the first command succeeded.

Hangman

grep actually implements a slightly modified version of regex for your search. So you can use regex.²³

- ▶ Download a dictionary text file.
- ▶ Search for words that match “kan _ _ roo”.
- ▶ Search for words starting with “tr”.

²³I won't cover regex.

Output redirection

Suppose you run a script that prints out a ton of data that you want to save. Instead of copying pasting the output, you can do the following.

```
$ neofetch --help > neofetch_help.txt
```

The standard output of “neofetch -help” is *piped* redirected to the file. ²⁴

²⁴There is another output reserved for “error” output that might still be output and not be in the file, in this case, use “&2>” instead.

Exercise

- ▶ You seem to vaguely remember a quote from one of Shakespeare's works.
- ▶ You only remember that the word "greatness" is in it.
- ▶ Your job is to use the Linux terminal to find the line with the word "greatness".²⁵
- ▶ You can download text file copies of Shakespeare's works here: <https://github.com/martin-gorner/tensorflow-rnn-shakespeare/tree/master/shakespeare>.

You can think of it as maybe you're in a giant code base and you're trying to find a certain term.

²⁵Yes, contrived example when you can Google the word with "Shakespeare". 

Hint

1. Download the repository. You can either “wget” the zip file and extract it, or use `git clone <repository>` to do so.
2. Use `grep --help`
3. `grep "greatness" ./docs` to list out all lines with that word in it.
4. You can try out other terms too, like maybe something about a “rose smelling sweet” or similar.

Further things you may want to learn about.

- ▶ Different types of shells (bash, zsh, fish, ...) and their differences.
- ▶ Shell expansions https://www.gnu.org/software/bash/manual/html_node/Shell-Expansions.html
- ▶ init daemons (systemctl, service, etc.)
- ▶ Desktop environments and/or window managers (KDE, Gnome, xfce, lxde, ...)
- ▶ Common command-line programs (sed, vim, tar, awk, find, ssh, etc.)
- ▶ File/Folder permissions.
- ▶ Shortcuts (Ctrl+C to stop current command, Ctrl+U to cut to start of line, Ctrl+R to search history from previous commands, etc.)
- ▶ Bash scripting.

Further resources.

- ▶ Linux command line Cheatsheet <https://cheatography.com/davechild/cheat-sheets/linux-command-line/>
- ▶ DigitalOcean introduction to Linux terminal <https://www.digitalocean.com/community/tutorials/an-introduction-to-the-linux-terminal>
- ▶ Bash (default shell) manual <https://www.gnu.org/software/bash/manual/>

Thank You!

- ▶ Thank You!
- ▶ Feedback form.
- ▶ Hope you will end up using Linux more!
- ▶ See you on campus!

