# Introduction to Linux and CLI

## Part 1

Francesco Ambrogi[1]

[1]**Assistant Professor**
Department of Mechanical and Materials Engineering
Queen's University

June 10, 2025

# Table of Contents



1. What is Linux
2. Basic UNIX concepts
3. The Shell
4. Basic shell commands
5. Installing new commands
6. Example 1

# How to follow this lecture



1. DO NOT take notes!
2. Open a terminal window and try to run the commands we will learn.
   - **Windows**: open a WSL terminal window.
   - **MacOS**: after starting the multipass instance, run **multipass shell openfoam**.
3. ASK QUESTIONS!

# The evolution of OSs



Linux is an operating system that evolved from a kernel created by Linus Torvalds when he was a student at the University of Helsinki (1991) [1]. It evolved from an old UNIX system, but:

- Linux **IS NOT** a program.
- Linux **is an interface** between computer/server hardware, and the programs which run on it.

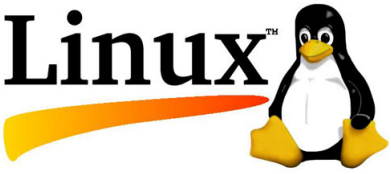


Figure: Linux Logo

[1] https://www.linux.org/threads/what-is-linux.4106/

# The kernel



People would often say "**Linux is just a kernel**". It turns out that a kernel is the focal point of any OS. A kernel is **what** tells the big chip what to do with the programs you are using. Example:

- The **dish** is your OS.
- The ingredients (tomato sauce, meat, parsley, etc) are the programs.
- The **pasta** is the **kernel**!

Figure: Pappardelle with wild boar ragu

# Why Linux???



Linux is JUST better:

- Linux is FREE and OPEN-SOURCE because of its community-based development.
- AMAZING software management through packages WHENEVER you want!
- Performance: Linux can run pretty much on anything, and it's very light and efficient.
- Stability and reliability (Internet servers, cloud systems, and clusters).
- **Safe** and **Transparent**!!

# Why NOT Linux???



Linux is NOT for everyone:

- Linux is NOT very user friendly. It can be quite confusing for non-tech people.
- Linux has small peripheral hardware drivers compared to windows.
- Compatibility issues.

# Basic UNIX concepts



1. **File**: data stored in a standard format that behaves in a certain way depending on its function in the system: **everything is a file in Linux**
2. **Program**: a file that can be executed (run).
3. **Process**: a program that is being executed.
4. **Ownership**: files/programs/processes are owned by a user and group.
5. **Hierachical Directory Structure**: files are organized in **directories** (folders) that can have a parent. Example: /home/user/sim1

Managing your files and processes is crucial for the correct utilization of the system!

# Shell, Terminal, Prompt



The **shell** is a program that takes (interpret) commands from the keyboard and gives them to the OS to perform. The shell is a command line interface (CLI). The shell provides:

- Built-in commands.
- Programming control structures.
- Environment variables.

Linux support several shells, but the default is usually BASH (**B**ourne **A**gain **SH**ell).

# Terminal



The **terminal** is a program called a **terminal emulator**. This is a program that opens a window and lets you interact with the shell. Once again, there are several terminal emulators supported by Linus systems: gnome-terminal, konsole, xterm, nxterm, and eterm.

# The prompt



```
francesco@rafaela : (~) ->
```

- **Username**: francesco
- **System/computer name**: rafaela
- $\sim$: current directory. In Linux the symbol $\sim$ is a shorthand for HOME directory.
- **Square**: input. The terminal is waiting for the user input.

# Give commands to the shell



The basic syntax to give commands to the Shell is the following:

command [OPTIONS] argument(s)

- **command**: is the keyword that identifies the specific command you are trying to run.
- **[OPTIONS]**: optional, they add more specifics to the command you are running.
- **argument(s)**: are the argument(s) required by the command to run.

# Print Working Directory (PWD)



In Linux the directory *you are currently* in is called **working directory**. When using the shell, it is very easy to get lost and the **pwd** command will be your compass.

```
francesco@rafaela : (07_MECH444) -> pwd
/Users/rafaela/QueensU/Teaching/07_MECH444
francesco@rafaela : (07_MECH444) ->
```

When we first log on to our Linux system, the working directory is set to our home directory ($\sim$). This is where we put our files. In most systems, the home directory will be called */home/username*, but it can be anything according to the wishes of the system administrator.

# List directory content (LS)



In Linux, to list the content of a directory, we use the command **ls**. The syntax is the following:

ls [option] [directory]

```
francesco@rafaela : (07_MECH444) -> ls
01_lectures 02_docs     Figures
francesco@rafaela : (07_MECH444) ->
```

Common options of the **ls** commands are:

- **ls -l**: List the files in a directory with extended information.
- **ls -a**: List all files, including hidden files (files with names starting with a dot, for example .bashrc).

# Change directory (CD)



In Linux, to move around directories (change) we use the command **change directory or CD**:

cd [directory]

```
francesco@rafaela : (07_MECH444) -> ls
01_lectures 02_docs     Figures
francesco@rafaela : (07_MECH444) -> cd 01_lectures/
francesco@rafaela : (01_lectures) -> ls
MECH444_intro_to_linux_cli.pdf
francesco@rafaela : (01_lectures) ->
```

the "." notation refers to the working directory itself and the ".." notation refers to the working directory's parent directory.

# Examples using CD



If I want to move "back" between direcotries:

```
francesco@rafaela : (01_lectures) -> pwd
/Users/rafaela/QueensU/Teaching/07_MECH444/01_lectures
francesco@rafaela : (01_lectures) -> cd ../
francesco@rafaela : (07_MECH444) -> pwd
/Users/rafaela/QueensU/Teaching/07_MECH444
francesco@rafaela : (07_MECH444) -> cd ../../
francesco@rafaela : (QueensU) -> pwd
/Users/rafaela/QueensU
francesco@rafaela : (QueensU) ->
```

# Create a file using TOUCH



In the shell, if you want to create a file (remember what a file is!) you can use the command **touch**. This command creates a file without opening it, and without adding anything on it.

```
francesco@rafaela : (01_lectures) -> ls
MECH444_intro_to_linux_cli.pdf
francesco@rafaela : (01_lectures) -> touch new_file.txt
francesco@rafaela : (01_lectures) -> ls
MECH444_intro_to_linux_cli.pdf new_file.txt
francesco@rafaela : (01_lectures) ->
```

# Keep in mind while creating files



**DANGER**: the name you choose for your files is VERY important.

- File names that begin with a period character are hidden. This only means that ls will not list them unless we say ls -a.
- File names in Linux, like Unix, are case-sensitive. The file names "File1" and "file1" refer to different files.
- **Do not** embed spaces in file names. If you want to represent spaces between words in a file name, use underscore characters. You will thank me later.

# Create a new directory using MKDIR



In the shell, if you want to create a new directory (remember what a directory is!) you can use the command **mkdir**. This command creates a new directory in the specified location with nothing in it. The syntax is the following:
**TASK**: I want to create a directory called "test_new_dir" inside "01_lectures"

```
francesco@rafaela : (07_MECH444) -> ls
01_lectures 02_docs     Figures
francesco@rafaela : (07_MECH444) -> mkdir 01_lectures/test_new_dir
francesco@rafaela : (07_MECH444) -> ls 01_lectures/
MECH444_intro_to_linux_cli.pdf new_file.txt                    test_new_dir
francesco@rafaela : (07_MECH444) ->
```

# A very important MAN



The **man** command, short for manual, is a powerful tool in the Linux operating system that allows users to access detailed information about various commands, utilities, and system calls. Just like any other shell command the syntax for the man command is:

man [OPTIONS] command

Manual pages pop up in the terminal itself, and can be quite overwhelming. They are, however, a GREAT resource! Here is how to navigate them:

- **Spacebar**: Move forward one page.
- **Enter**: Move forward one line.
- **B**: Move backward one page.
- **Q**: Quit the manual viewer.

# The very useful TREE command



In Linux/UNIX is very easy to get lost, especially if one is used to a GUI. The **tree command** is a recursive directory listing program that produces a depth-indented listing of files. With no arguments, the tree lists the files in the current directory. You can install TREE by running:

sudo apt install tree

```
francesco@rafaela : (07_MECH444) -> ls
01_lectures 02_docs     Figures
francesco@rafaela : (07_MECH444) -> tree 01_lectures/
01_lectures/
├── MECH444_intro_to_linux_cli.pdf
└── new_file.txt

1 directory, 2 files
francesco@rafaela : (07_MECH444) ->
```

# End of class example



Open a new terminal window and perform the following:

1. Go to your HOME directory.
2. Create a new directory called "intro_shell"
3. Move into the new directory.
4. Create an empty .txt file called "my_first_file.txt"
5. List the content of your new directory.