

Open source scientific computing language and reproducible computational infrastructure towards project-centric Omics Data Integration

19th to 22nd November 2025

Dr. B. R. Ambedkar Centre for Biomedical Research
University of Delhi, North Campus, Delhi-110007

Hands on Session

19/11/2025

Teaching Fellows:

Vinita Sharma

PhD Student (Systems Medicine)

Department of Biology

Ashoka University

Gautam Ahuja

Multimodal AI Engineer

Koita Centre for Digital Health at Ashoka (KCDH-A)

Ashoka University



ashoka
UNIVERSITY

Session Goals

Gain a foundational understanding of open source computing tools and infrastructure for effective data insights generation

- Scientific Computing Tools and Languages
- Unix-based operating system
- Windows Subsystem for Linux
- Conda environments and basics of Python
- Jupyter Notebooks and Colab

Scientific Computing

Scientific Computing Languages and Tools

Programming Languages

Low Level: Provides little or no abstraction from a computer's instruction set architecture.
Examples:

- Rust
- C/C++
- Assembly
- Fortran
- COBOL
- Haskell

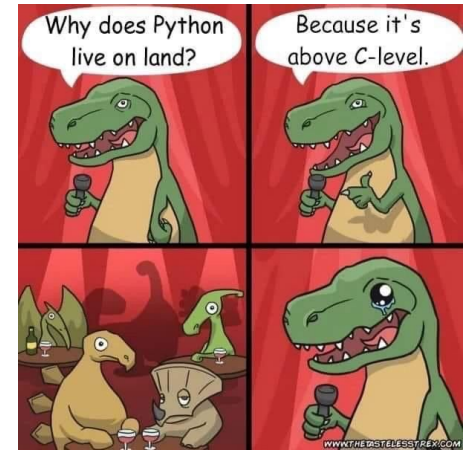


Scientific Programming

High Level: Easier for humans to understand and write. Examples:

- Python
- R
- Julia

Scientific programming languages are built on top of low-level programming languages.



Tools

- **Unix-based Operating System:** Many data science tools and libraries work best in Linux environments
- **Environment Management System:**
Miniconda / Anaconda: Founded in 2012 by Peter Wang and Travis Oliphant out of the need to bring Python into business data analytics. It has a Simplified Package Management (conda).



Tools



- **OS-Level Virtualization Systems:**
 - Docker: A platform designed to help developers build, share, and run container applications. We handle the tedious setup, so you can focus on the code.
- **Version Control & Other tools:**
 - Git



Version Control – Git

- Git is a distributed version control system that allows developers to track changes in code and collaborate with others.

- **Version control**

Git allows for easy version control, making it easy to keep track of changes and revert to previous versions if needed.

- **Collaboration**

Git allows multiple developers to work on the same project without overwriting each other's code. Changes can be easily merged and conflicts can be resolved.

- **Branching and merging**

Git makes it easy to create different branches for different features or experiments, and then merge them back into the main codebase when they're ready.

- **Offline work**

Git allows developers to work offline and then sync their changes when they're back online, making it easy to work from anywhere without worrying about internet connectivity.

Git Workflow

To outline the basic Git workflow



git clone [repo URL]

Create a copy of a remote repository on your local machine



git branch [branch name]

Create a new branch for your changes



git commit -m [commit message]

Save changes to your local repository



git push [remote] [branch]

Upload your changes to a remote repository



git pull [remote] [branch]

Download changes from a remote repository to your local repository



git merge [branch name]

Combine changes from one branch into another

Typical GitHub Workflow

- Make Changes in the code
- `git add <file-name>`
- `git commit -m "Message"`
- `git push`

Also Helpful:

- `git status`
- `git reset / git reset hard (BE CAUTIOUS)`
- `git revert`

Resources

Additional resources for learning Git and GitHub

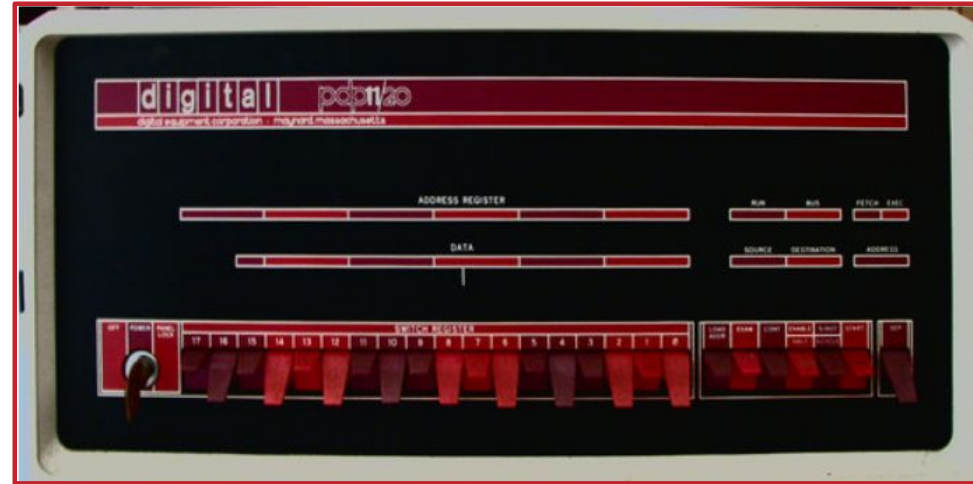
Git Documentation	https://git-scm.com/doc
GitHub Learning Lab	https://lab.github.com/
GitHub Guides	https://github.com/git-guides
Pro Git Book	https://git-scm.com/book/en/v2

Unix

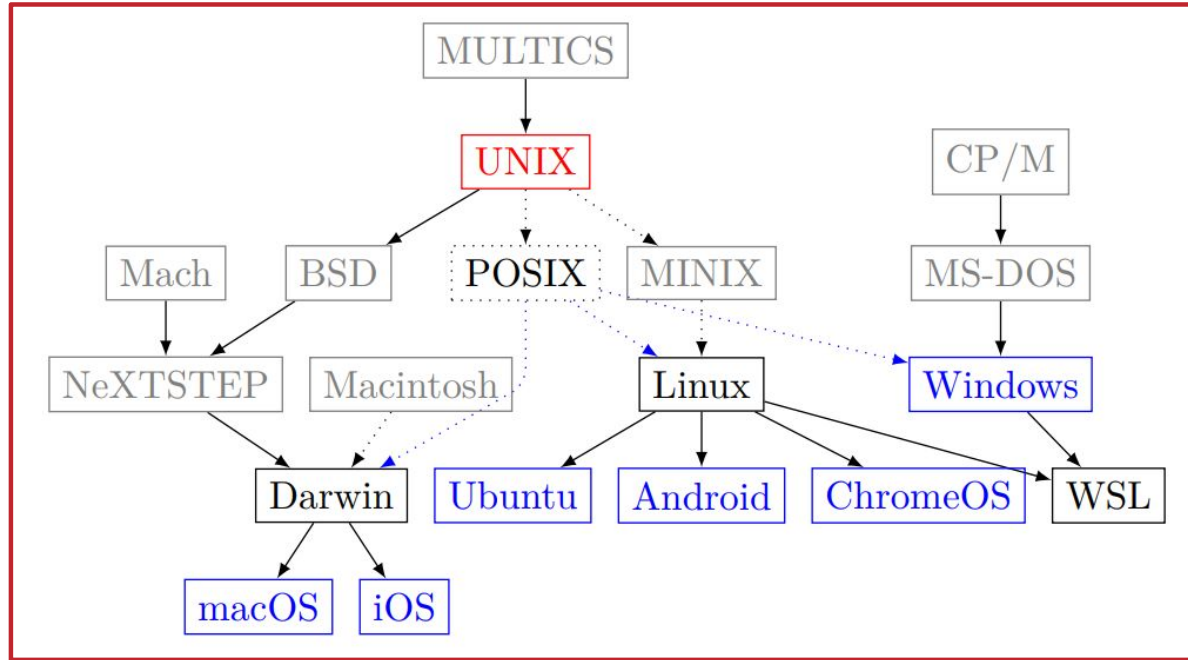
Introduction to Unix based operating system (linux) and linux shell

What is UNIX?

- Originally developed in 1969 at AT&T Bell Labs
- [Ken Thompson and Dennis Ritchie Explain UNIX \(Bell Labs\)](#)



What is UNIX?



Almost all modern computing is derived from the legacy of UNIX

What is Linux?

- Linux is a “UNIX-Like” operating system
- The operating system manages the relationship between computer hardware, software resources and user interaction points
- Unix-like computer operating system assembled under the model of free and open-source software development and distribution.
- Comes in several “distributions” to serve different purposes



What is Linux?

- Linux kernel was written by Linus Torvalds using the GNU Project open source system in 1991
- GNU Project believed in freedom
 - freedom to run a program for any purpose
 - freedom to study the mechanics of the program and modify it
 - freedom to redistribute copies
 - freedom to improve and change modified versions for public use



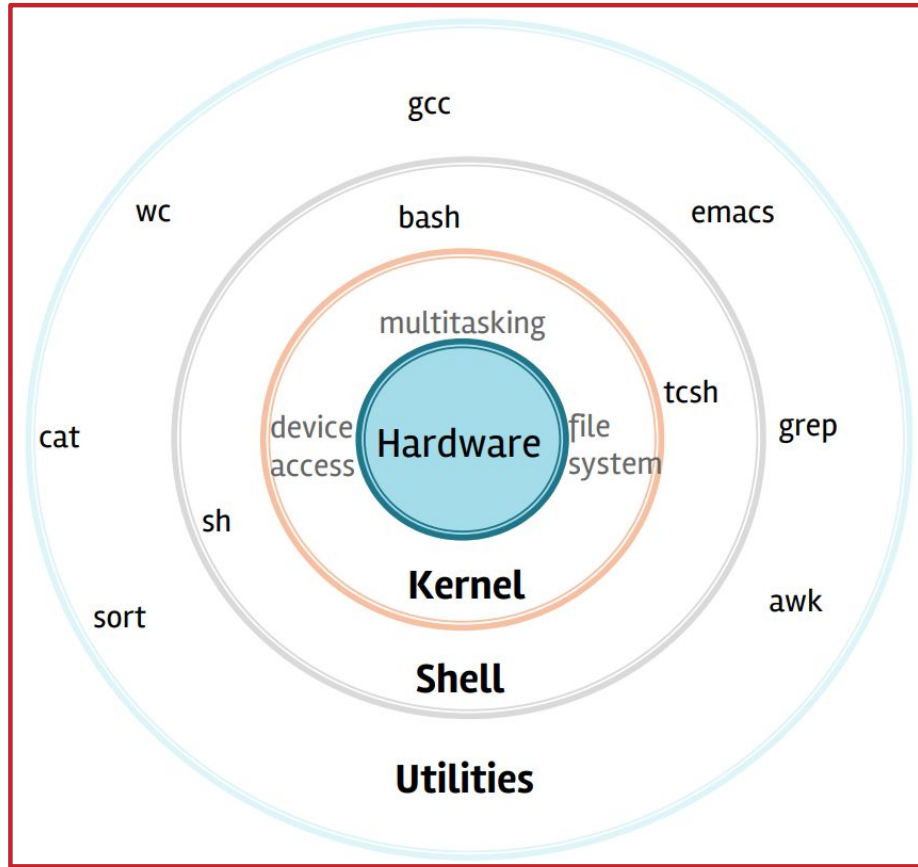
Linus Torvalds



A set of programs written by Richard Stallman and others. They are the GNU utilities.

What is Linux?

- Bird's eye view
- “Rings of Privilege”



Where is Linux?

- World Wide Web
 - 67% of the world's web-servers run Linux (2016)
- Research/High-Performance Compute
 - Google, Amazon, NSA, 100% of TOP 500 Supercomputers.
- Modern Smartphones and devices
 - The Android phone
 - Amazon Kindle
 - Smart TVs/Devices

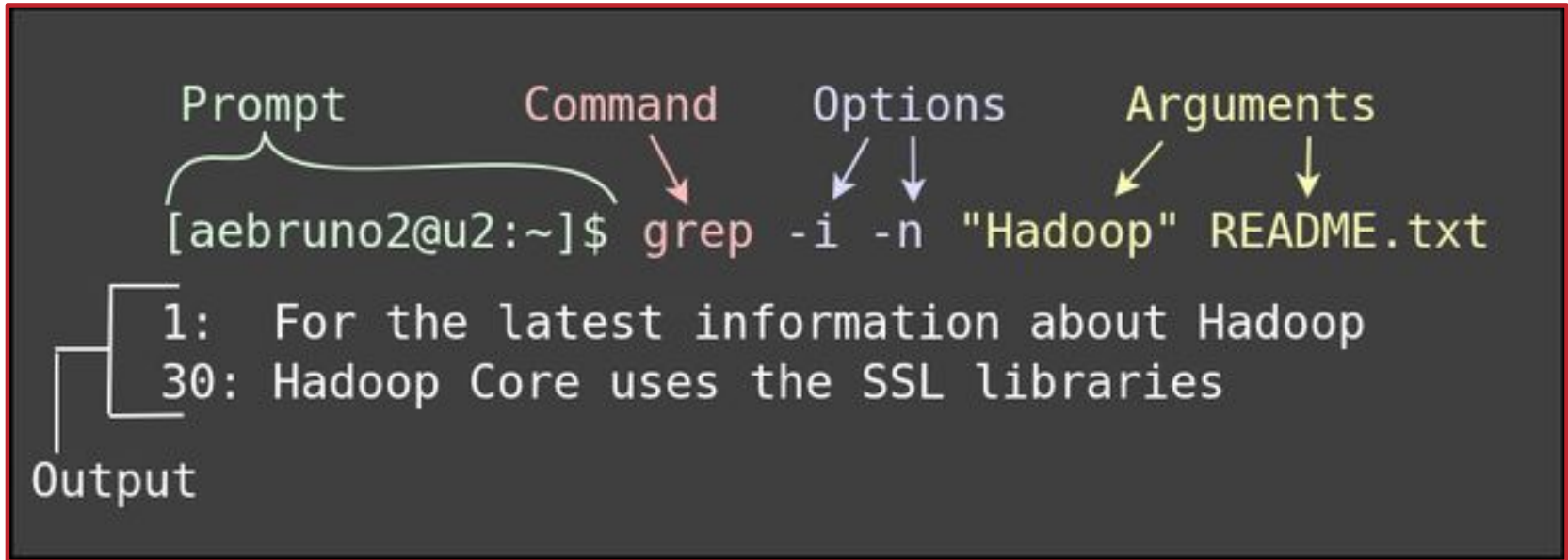
Windows Subsystem for Linux (WSL)

- It is a way to run a Linux environment and applications on a Windows system without having to boot out of Windows.
- WSL is capable of running many different Linux based operating systems
- Lets developers run a GNU/Linux environment -- including most command-line tools, utilities, and applications -- directly on Windows, unmodified, without the overhead of a traditional virtual machine
- **[August 2025] Windows Subsystem for Linux is now Open Source!**

The UNIX Shell

- Operating systems have user interfaces:
 - GUI - Graphical User Interface
 - CLI - Command Line Interface
- CLI - Command Line Interface
 - Control the computer with commands
 - “Shell” - a program that accepts commands and performs operating system functions
 - We’ll be using a shell called BASH

How to read the SHELL



LET US TRY SOME COMMANDS

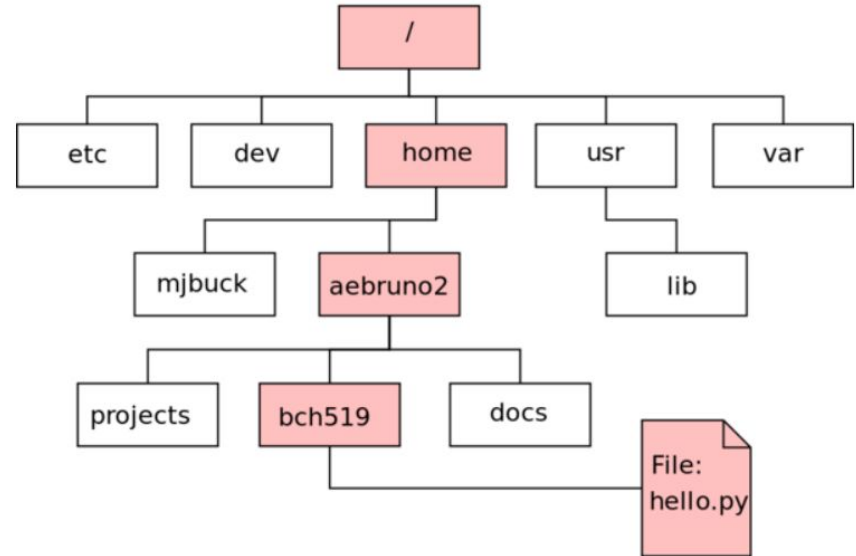
- date
- echo
- whoami
- man
- grep
- cat
- head
- tail
- ps
- top
-

Files, Directories, and Paths

- Basic unit is a **File**
- Files are organized in directories (folders)
- Navigate filesystem using paths

Hierarchical Filesystem

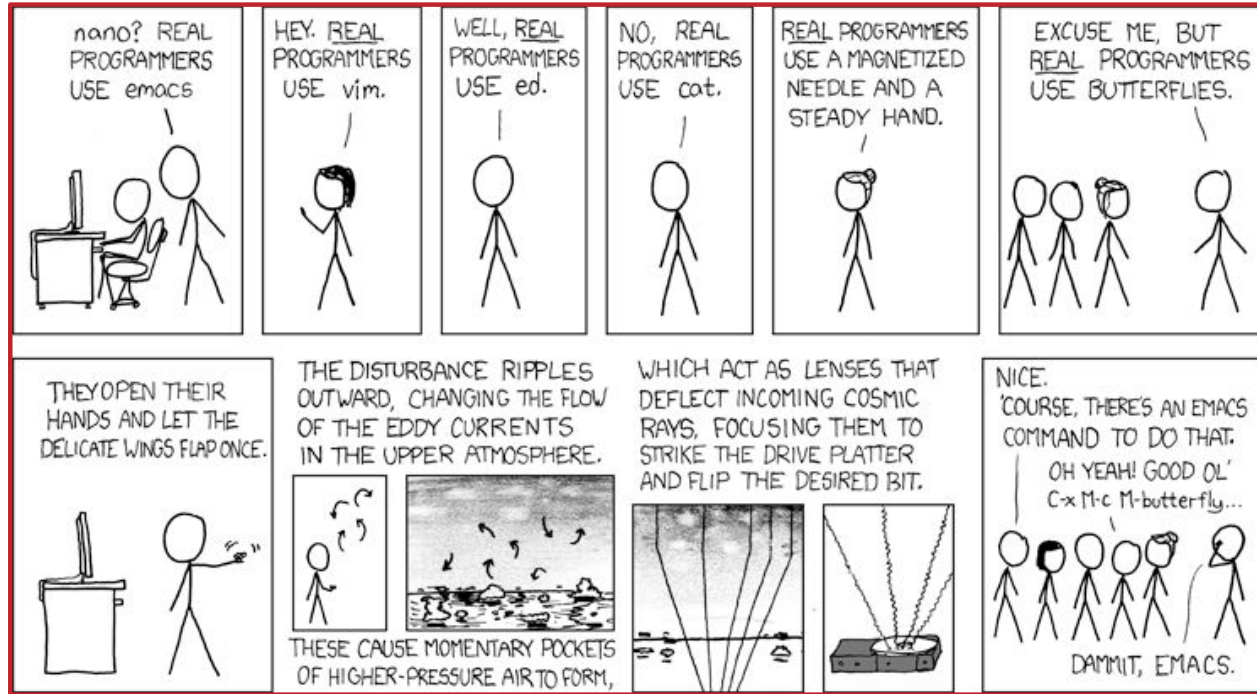
/home/aebruno2/bch519/hello.py



Editors

- nano
- emacs
- vi / vim

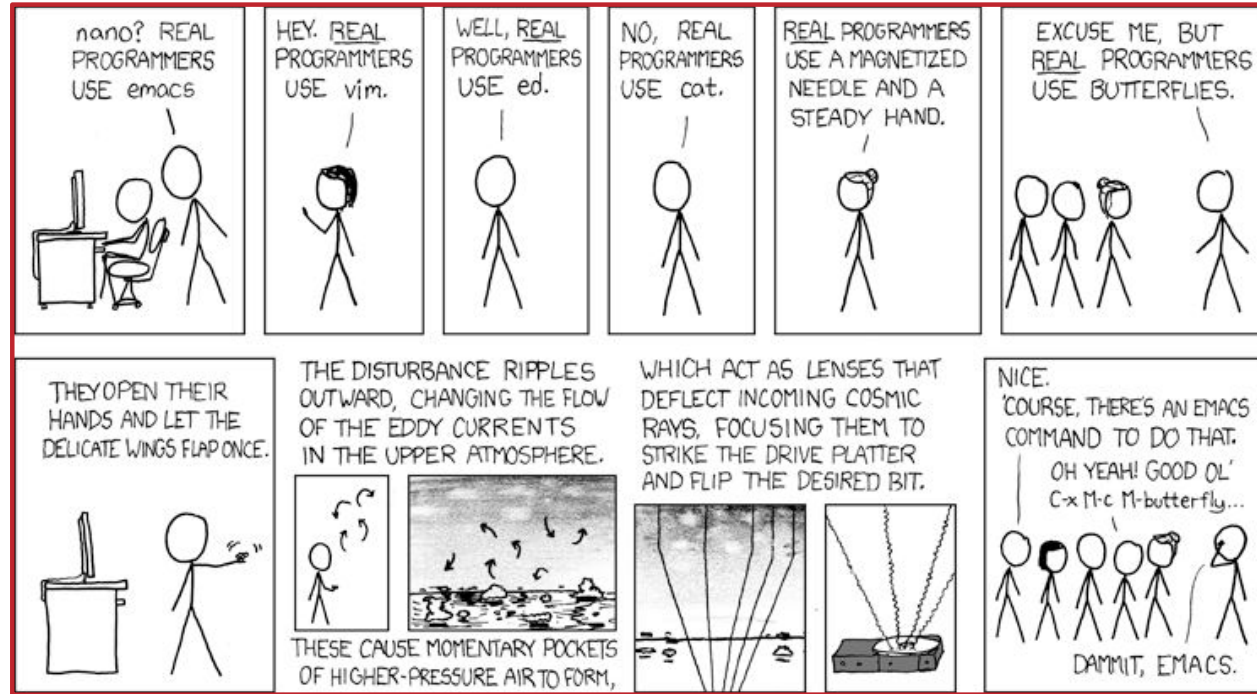
Fun:
[Interview with an Emacs Enthusiast](#)
[\[Colorized\]](#)



Editors

- CLI
 - nano
 - emacs
 - vi / vim
- VSCode!

Fun:
[Interview with an Emacs Enthusiast](#)
[\[Colorized\]](#)



LET US TRY SOME COMMANDS

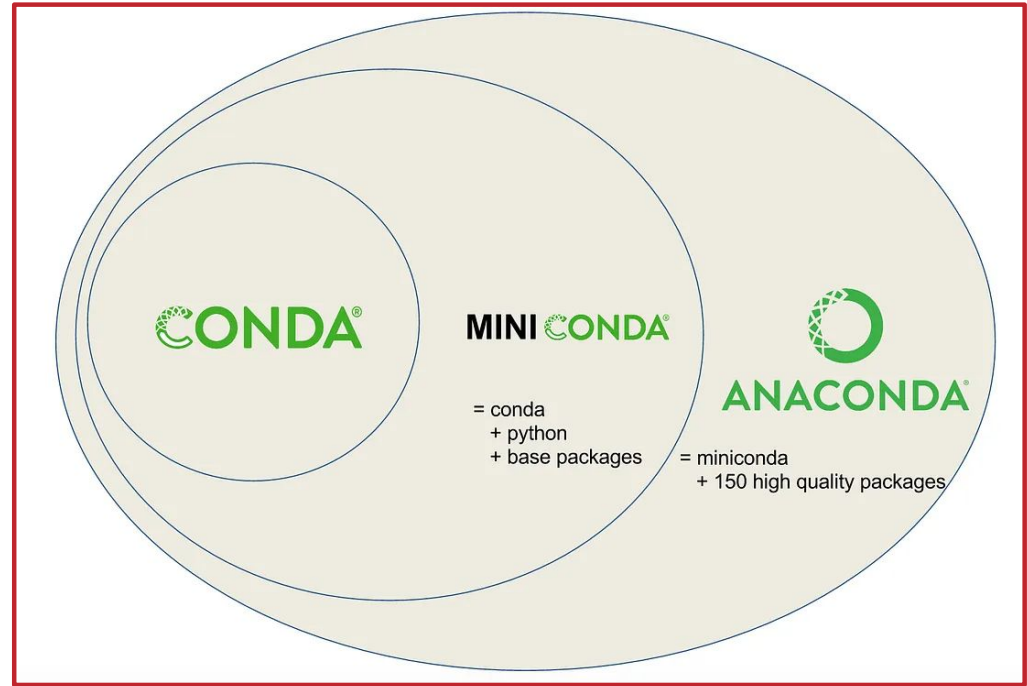
- ls
- pwd
- cd
- touch
- cp
- mv
- rm
- wc -l

Environment Manager

Introduction to Conda

What is CONDA?

- Conda is an open source package and environment management system that runs on Windows, Mac OS and Linux.
- Use case: if you need a package that requires a different version of Python.



Why should you use Conda?

- Conda provides prebuilt packages or binaries (which generally avoids the need to deal with compiling packages from source)
- Conda is cross platform, with support for Windows, MacOS, GNU/Linux, and support for multiple hardware platforms, such as x86 and Power 8 and 9.
- Where a library or tools is not already packaged for install using conda, Conda allows for using other package management tools (such as pip) inside Conda environments.

Getting Started

- **Create a new environment:** `conda create -n acbr-ws-25`
- **Activate new environment:** `conda activate acbr-ws-25`
- **Install python:** `conda install python=3.12`
- **Install package:** `conda install pandas numpy matplotlib`
- **Deactivate environment:** `conda deactivate`
- **Remove an environment:** `conda remove -n acbr-ws-25 --all`

Towards our first python program

- **Activate new environment:** `conda activate acbr-ws-25`
- **In command line type:** `python3`
- **Now:** `>>print("Hello World")`

GitHub, Notebooks and Colab

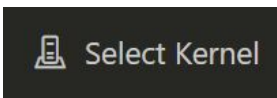
Introduction to Jupyter Notebooks and Version Control

Jupyter Notebooks – Python

- The default python interpreter processes only Python source code
- Jupyter notebooks allow us to run python code in an interactive environment.
 - The file extension is: .ipynb which stands for interactive python notebook
- `pip install ipython ipykernel`
- Using VSCode to run jupyter notebooks
- You can also run them on [Google Colab](#)!
 - Google Colab is a free, hosted Jupyter Notebook service that allows you to write and execute Python code in your browser

Jupyter Notebooks – R

- Conda environment
 - `conda create -n acbr-ws-25-r`
 - `conda activate acbr-ws-25-r`
 - `conda install conda-forge::r-base`
 - `conda install ipykernel`
- Install VScode extensions - jupyter, R (left sidebar)
- Open R
 - `install.packages("IRkernel")`
 - `IRkernel::installspec()`
- Relaunch VScode
 - Create a `.ipynb` file
 - Select kernel [top right]



Python Booster

- Variables
- Data Types
 - Numbers: Integers and Floats
 - Strings & String operations
 - Boolean
- Data Structures
 - Tuples & Arrays
 - Dictionaries
- Loops & Functions

Q & A

Any queries?