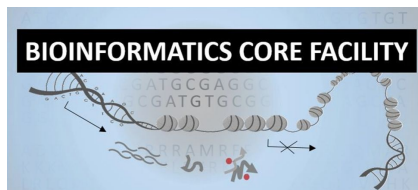


# Welcome to ABC.4

3rd September 2024  
abc.au.dk



Health  
Data Science  
Sandbox

# Agenda

- Introduction to the Terminal and the Bash shell
- Strong points of the bash shell
- Commands Syntax, File system and important things to know
- Hands-On Exercises or Open Coding and support

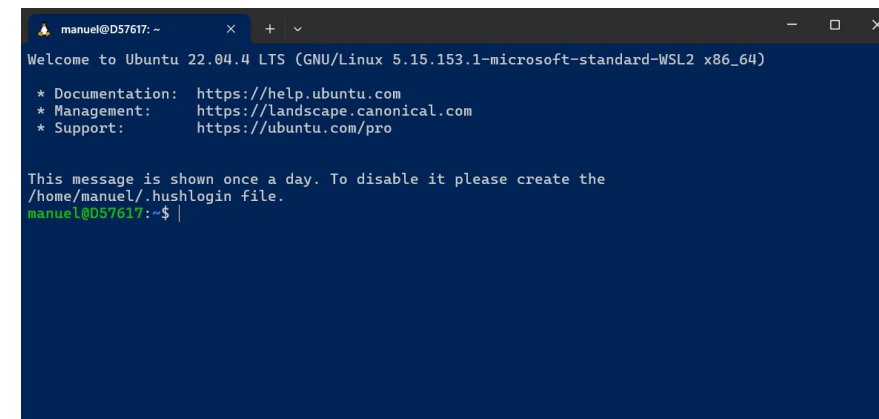
# Concepts

**Terminal**: A software environment for running text commands via the command-line interface (CLI).

Examples:

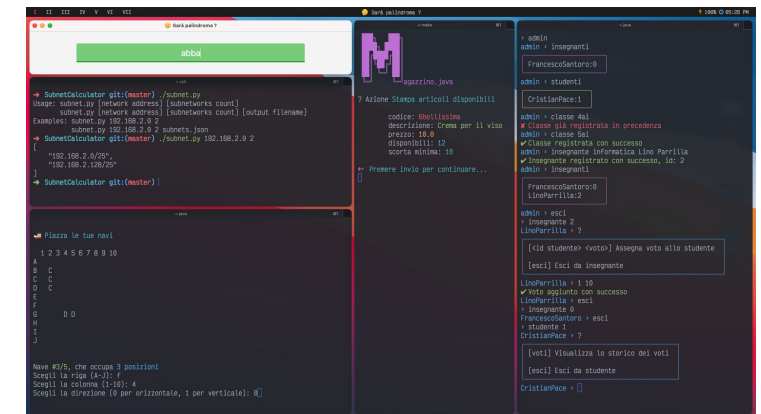
- Windows: Windows Terminal, PowerShell, Command Prompt.
- Linux: Terminal, Konsole, Terminator.
- MacOS: Terminal, iTerm2.

**Shell**: A text-based language and interpreter that interacts with the operating system, executing commands you input in the terminal.



```
manuel@D57617: ~  
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 5.15.153.1-microsoft-standard-WSL2 x86_64)  
  
* Documentation: https://help.ubuntu.com  
* Management:   https://landscape.canonical.com  
* Support:       https://ubuntu.com/pro  
  
This message is shown once a day. To disable it please create the  
/home/manuel/.hushlogin file.  
manuel@D57617:~$
```

Windows terminal



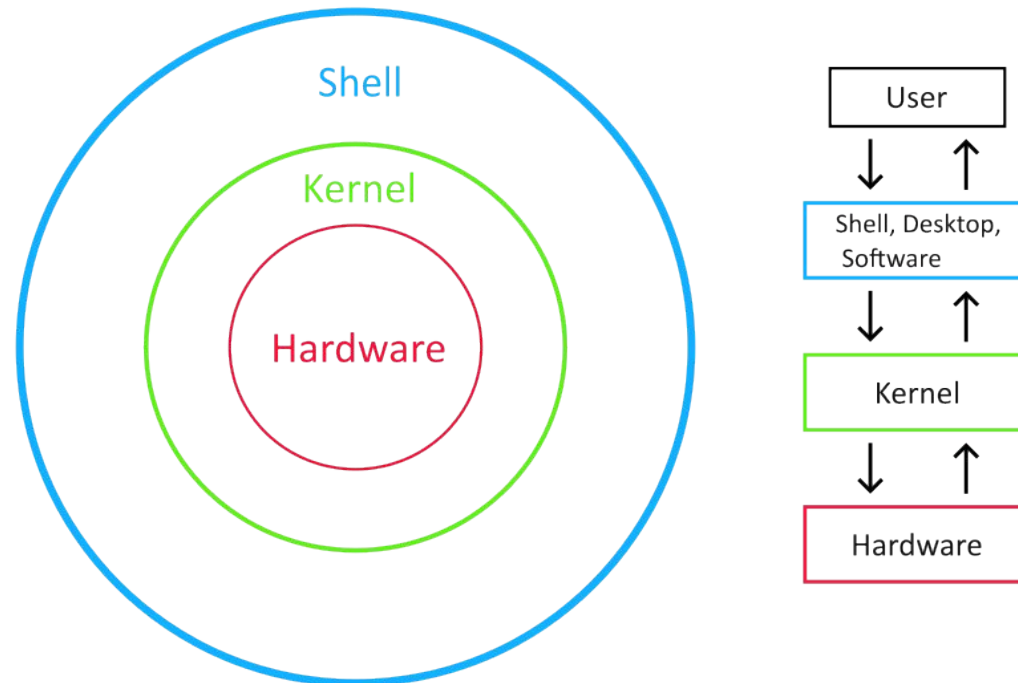
iTerm2



Bash command-line shell

# What is Bash and how it works?

- Bash **shell** is a **text-based language (and an interpreter)** that processes commands typed into the CLI.
- it **interprets (translates)** for the **kernel** the bash commands
- The kernel **orchestrates** the **hardware** resources to run the commands

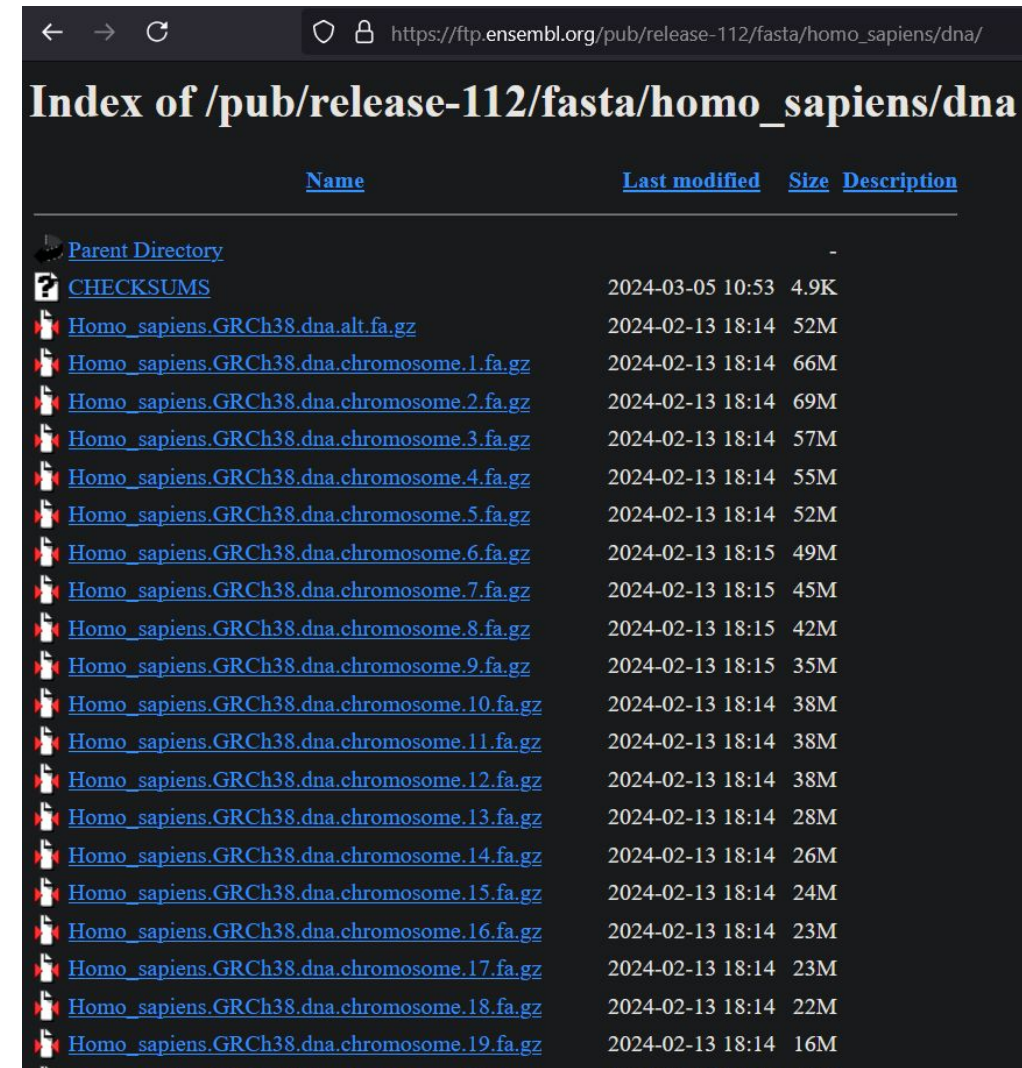


# What Makes Bash Worth Learning?

- **Automate** repetitive tasks
- Has **many bioinformatics tools** available
- The preferred **language for computing clusters** (e.g., GenomeDK)
- **Handles** many & large files
- Practical for creating **pipelines**

Example: downloading all files in a directory:

```
wget -m  
ftp://ftp.ensembl.org/pub/release-112/fasta/homo_sapiens/dna/
```

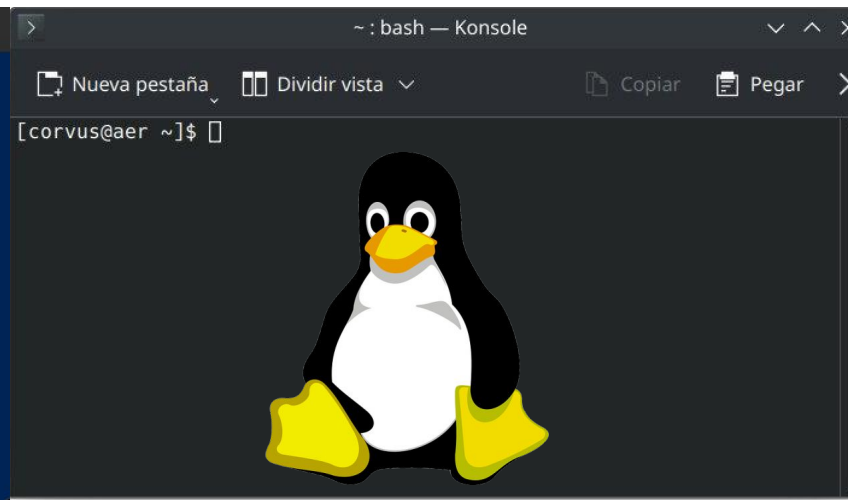
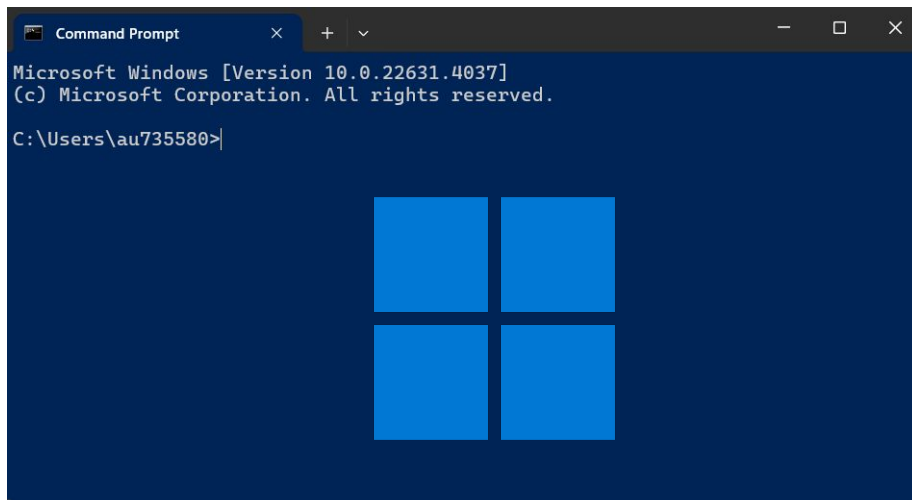


Index of /pub/release-112/fasta/homo\_sapiens/dna/

Name	Last modified	Size	Description
<a href="#">Parent Directory</a>	-	-	-
<a href="#">CHECKSUMS</a>	2024-03-05 10:53	4.9K	
<a href="#">Homo_sapiens.GRCh38.dna.alt.fa.gz</a>	2024-02-13 18:14	52M	
<a href="#">Homo_sapiens.GRCh38.dna.chromosome.1.fa.gz</a>	2024-02-13 18:14	66M	
<a href="#">Homo_sapiens.GRCh38.dna.chromosome.2.fa.gz</a>	2024-02-13 18:14	69M	
<a href="#">Homo_sapiens.GRCh38.dna.chromosome.3.fa.gz</a>	2024-02-13 18:14	57M	
<a href="#">Homo_sapiens.GRCh38.dna.chromosome.4.fa.gz</a>	2024-02-13 18:14	55M	
<a href="#">Homo_sapiens.GRCh38.dna.chromosome.5.fa.gz</a>	2024-02-13 18:14	52M	
<a href="#">Homo_sapiens.GRCh38.dna.chromosome.6.fa.gz</a>	2024-02-13 18:15	49M	
<a href="#">Homo_sapiens.GRCh38.dna.chromosome.7.fa.gz</a>	2024-02-13 18:15	45M	
<a href="#">Homo_sapiens.GRCh38.dna.chromosome.8.fa.gz</a>	2024-02-13 18:15	42M	
<a href="#">Homo_sapiens.GRCh38.dna.chromosome.9.fa.gz</a>	2024-02-13 18:15	35M	
<a href="#">Homo_sapiens.GRCh38.dna.chromosome.10.fa.gz</a>	2024-02-13 18:14	38M	
<a href="#">Homo_sapiens.GRCh38.dna.chromosome.11.fa.gz</a>	2024-02-13 18:14	38M	
<a href="#">Homo_sapiens.GRCh38.dna.chromosome.12.fa.gz</a>	2024-02-13 18:14	38M	
<a href="#">Homo_sapiens.GRCh38.dna.chromosome.13.fa.gz</a>	2024-02-13 18:14	28M	
<a href="#">Homo_sapiens.GRCh38.dna.chromosome.14.fa.gz</a>	2024-02-13 18:14	26M	
<a href="#">Homo_sapiens.GRCh38.dna.chromosome.15.fa.gz</a>	2024-02-13 18:14	24M	
<a href="#">Homo_sapiens.GRCh38.dna.chromosome.16.fa.gz</a>	2024-02-13 18:14	23M	
<a href="#">Homo_sapiens.GRCh38.dna.chromosome.17.fa.gz</a>	2024-02-13 18:14	23M	
<a href="#">Homo_sapiens.GRCh38.dna.chromosome.18.fa.gz</a>	2024-02-13 18:14	22M	
<a href="#">Homo_sapiens.GRCh38.dna.chromosome.19.fa.gz</a>	2024-02-13 18:14	16M	

# Why not alternative shells?

- Many programmers and systems **don't use Bash by default** (e.g., *CMD* in Windows, *zsh* alternative in Mac, *fish* command line, ...)
- We focus on **Bash** because it's powerful and basically the **standard in bioinformatics**.
- Cool thing: **Windows users can run Bash** via Windows Subsystem for Linux (**WSL**).



# How do I learn all these commands?

Commands are often easy to remember thanks to their **mnemonic nature**, such as **ls**, **cd**, **mkdir**, **cp**, and **rm**.

Help page

```
manuel@D57617:~$ grep --help
Usage: grep [OPTION]... PATTERNS [FILE]...
Search for PATTERNS in each FILE.
Example: grep -i 'hello world' menu.h main.c
PATTERNS can contain multiple patterns separated by newlines.

Pattern selection and interpretation:
-E, --extended-regexp  PATTERNS are extended regular expressions
-F, --fixed-strings    PATTERNS are strings
-G, --basic-regexp     PATTERNS are basic regular expressions
-P, --perl-regexp      PATTERNS are Perl regular expressions
-e, --regexp=PATTERNS  use PATTERNS for matching
-f, --file=FILE        take PATTERNS from FILE
-i, --ignore-case       ignore case distinctions in patterns and data
                        --no-ignore-case do not ignore case distinctions (default)
-W, --word-regexp       match only whole words
-x, --line-regexp       match only whole lines
-z, --null-data         a data line ends in 0 byte, not newline
```

grep(1) — Linux manual page

NAME | SYNOPSIS | DESCRIPTION | OPTIONS | REGULAR EXPRESSIONS | EXIT STATUS | ENVIRONMENT | NOTES | COPYRIGHT | BUGS | EXAMPLE | SEE ALSO | COLOPHON

Search online pages

grep(1) User Commands GREP(1)

NAME top

grep - print lines that match patterns

SYNOPSIS top

grep [OPTION...] PATTERNS [FILE...]  
grep [OPTION...] -e PATTERNS ... [FILE...]  
grep [OPTION...] -f PATTERN\_FILE ... [FILE...]

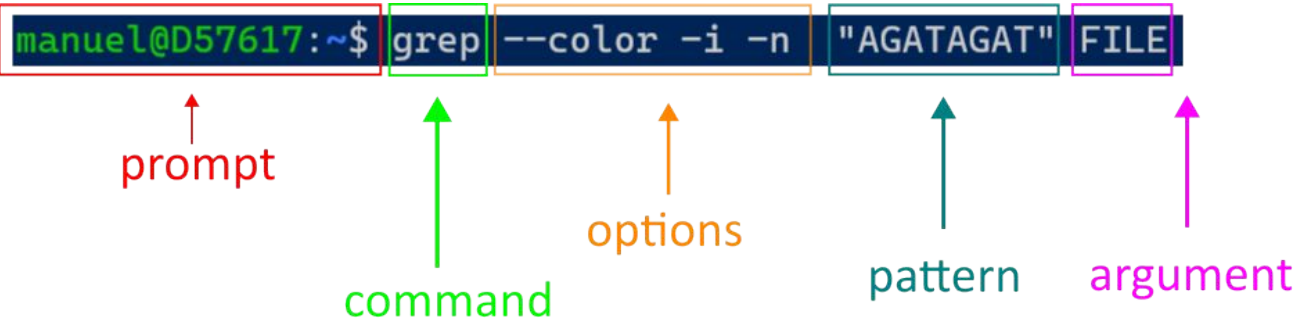
DESCRIPTION top

grep searches for PATTERNS in each FILE. PATTERNS is one or more patterns separated by newline characters, and grep prints each line that matches a pattern. Typically PATTERNS should be quoted when grep is used in a shell command.  
  
A FILE of "-" stands for standard input. If no FILE is given

man page

(e.g. online search for *grep man*)

## Anatomy (syntax) of a command

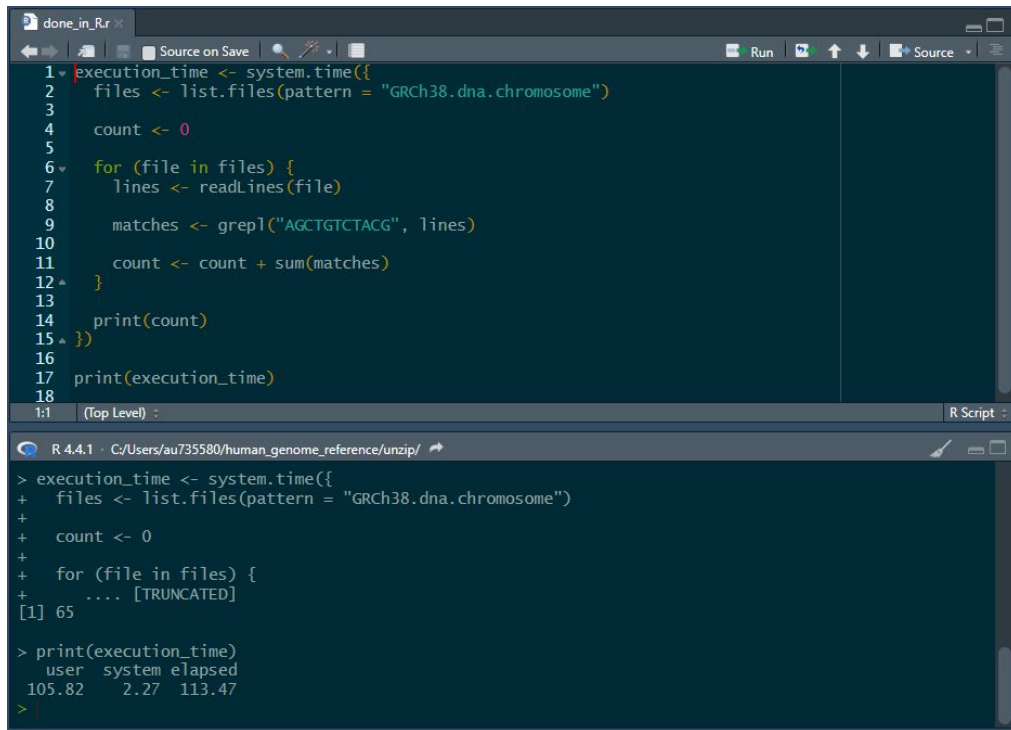


Google



# Bash: Efficiency in Simplicity

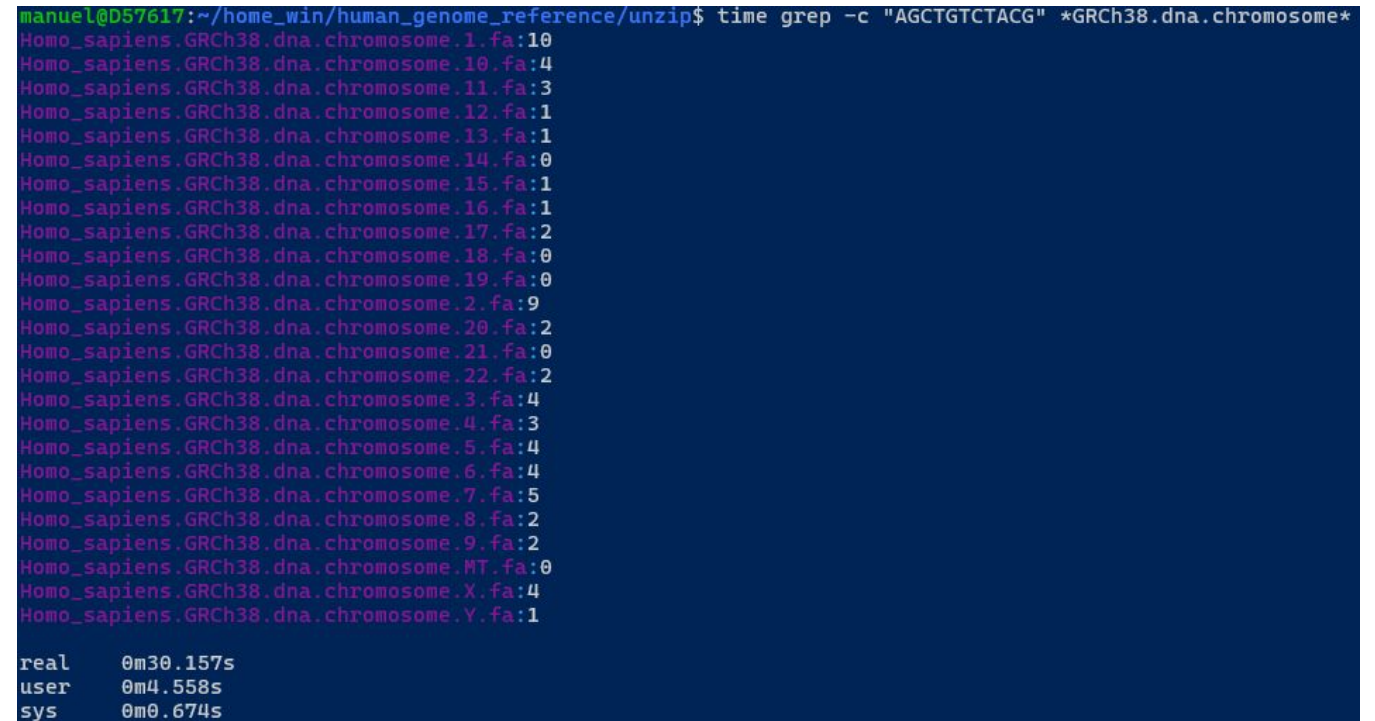
Compare the time taken to search for the sequence "AGCTGTCTACG" across the human genome (25 files, 2.92 GB).



```
1 execution_time <- system.time({
2   files <- list.files(pattern = "GRCh38.dna.chromosome")
3
4   count <- 0
5
6   for (file in files) {
7     lines <- readLines(file)
8     matches <- grep("AGCTGTCTACG", lines)
9     count <- count + sum(matches)
10  }
11  print(count)
12 })
13
14 print(execution_time)
15
16
17
18
```

```
> execution_time <- system.time({
+   files <- list.files(pattern = "GRCh38.dna.chromosome")
+   count <- 0
+   for (file in files) {
+     .... [TRUNCATED]
+   }
+ })
[1] 65
> print(execution_time)
  user system elapsed
105.82   2.27  113.47
>
```

R Script: 113 seconds / 9 lines of code



```
manuel@DS7617:~/home_win/human_genome_reference/unzip$ time grep -c "AGCTGTCTACG" *GRCh38.dna.chromosome*
Homo_sapiens.GRCh38.dna.chromosome.1.fa:10
Homo_sapiens.GRCh38.dna.chromosome.10.fa:4
Homo_sapiens.GRCh38.dna.chromosome.11.fa:3
Homo_sapiens.GRCh38.dna.chromosome.12.fa:1
Homo_sapiens.GRCh38.dna.chromosome.13.fa:1
Homo_sapiens.GRCh38.dna.chromosome.14.fa:0
Homo_sapiens.GRCh38.dna.chromosome.15.fa:1
Homo_sapiens.GRCh38.dna.chromosome.16.fa:1
Homo_sapiens.GRCh38.dna.chromosome.17.fa:2
Homo_sapiens.GRCh38.dna.chromosome.18.fa:0
Homo_sapiens.GRCh38.dna.chromosome.19.fa:0
Homo_sapiens.GRCh38.dna.chromosome.2.fa:9
Homo_sapiens.GRCh38.dna.chromosome.20.fa:2
Homo_sapiens.GRCh38.dna.chromosome.21.fa:0
Homo_sapiens.GRCh38.dna.chromosome.22.fa:2
Homo_sapiens.GRCh38.dna.chromosome.3.fa:4
Homo_sapiens.GRCh38.dna.chromosome.4.fa:3
Homo_sapiens.GRCh38.dna.chromosome.5.fa:4
Homo_sapiens.GRCh38.dna.chromosome.6.fa:4
Homo_sapiens.GRCh38.dna.chromosome.7.fa:5
Homo_sapiens.GRCh38.dna.chromosome.8.fa:2
Homo_sapiens.GRCh38.dna.chromosome.9.fa:2
Homo_sapiens.GRCh38.dna.chromosome.MT.fa:0
Homo_sapiens.GRCh38.dna.chromosome.X.fa:4
Homo_sapiens.GRCh38.dna.chromosome.Y.fa:1

real    0m30.157s
user    0m4.558s
sys     0m0.674s
```

Bash: 30 seconds / 1 line of code

This is **the Unix philosophy**:

Programs that do **one thing** and do it **well**.  
Programs to work together. Programs to handle  
text streams, because that is a universal interface.”

Doug McIlory

A few lines in R/Python



A short line in Bash



# Understanding the Unix File System

- **Root directory /**
  - Top-level of the system
  - Contains all the files and directories
- **Home directory ~**
  - Your **private folder** in the UNIX system
- **Navigating the File system**
  - Display your **current working directory (CWD)** with **pwd**
  - Paths on the file system
    - **Relative path** starts from the CWD
    - **Absolute path** starts from the root

Documents

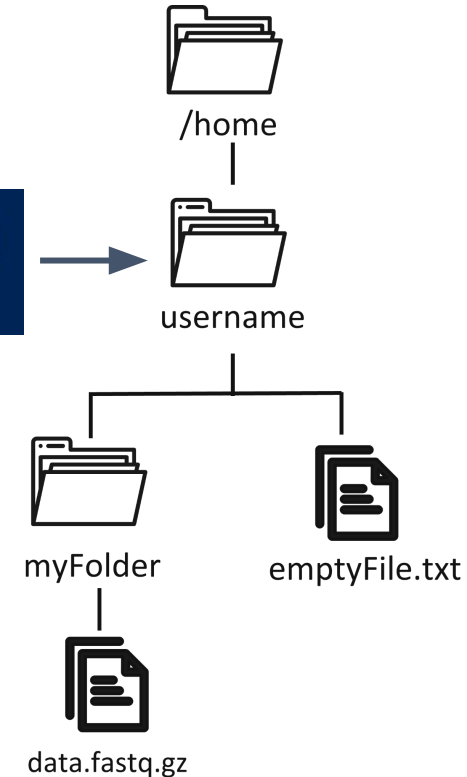
Relative path

=

/home/username/Documents

Absolute path

```
manuel@D57617:~$ pwd  
/home/manuel
```



Always be aware of your CWD in the file system!

# Some basic points to always keep in mind!

- **Avoid spaces** in file names; use
  - underscores `_` (**snake case**),
  - hyphens `-` (**kebab case**), or
  - **camelCase** instead.
  - Example: `newProjectData`, `new_project_data`
- **Don't start names with numbers**, as this can cause issues with some commands or softwares.
- Deleted files **are always unrecoverable**. No trash bin in bash shell.
- Also **overwriting** a file makes the old one unrecoverable.
- **Always think twice** when removing

```
manuel@D57617:~$ mv report 2024.pdf final reports/  
mv: target 'reports/' is not a directory  
manuel@D57617:~$
```

# Getting Started with the Terminal

- Becoming proficient with the terminal takes **practice**, but it's a valuable skill you will probably use in your career. Make use of **man pages and online resources!** Start right away learning the basics:

`abc.au.dk` → Documentation → ABC4 tutorial

[Link](#)