Basic command line usage in the Bash shell

Allan van Hulst

Getting started

You may participate using one of the following possibilities:

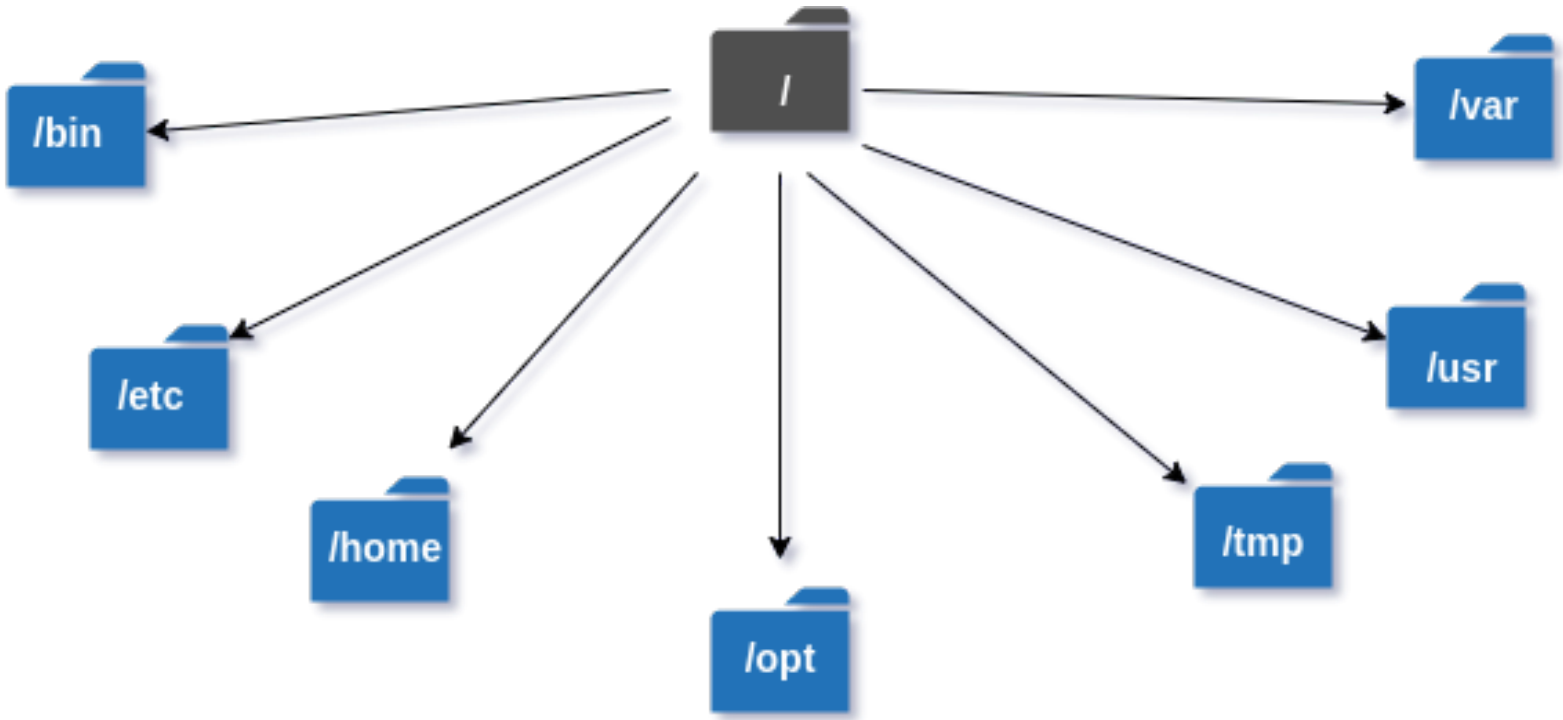
- By having brought your own system (Linux or MacOS)
- By having online access to a VM via SSH (e.g. SURF Research Cloud)
- By setting up an environment for experimentation as shown on the next slide

How to get an experimental environment for Bash

1. Go to:
   `https://open-science-cloud.ec.europa.eu/dashboard/notebooks`
2. Login with your institutional credentials
3. Go to: File Sync and Share to obtain storage
4. Go to: Interactive Notebooks and click 'Get Access'
5. Click on 'View Externally' once the notebook has started
6. Click on 'Terminal'

Part I: The Essentials

Directory structure in a Linux environment



Navigating the directory structure

- You are always working in a unique working directory
- Use the command **cd** to change the current working directory
  - Example: **cd test** changes the working directory to **test**
- If you login to a system then the initial working directory is your home directory
- Paths to files are interpreted relative to the current working directory

Modifying the directory structure

- Use **mkdir** to create a new directory
  - Example: **mkdir research** creates a new directory **research** in the current directory
- Use **rmdir** to remove a directory
  - Important: a directory needs to be empty otherwise it cannot be removed

Files in a directory

- Use the command **ls** to list the contents of a directory
- The command **ls** has many options
- If no pathname is given as an option then **ls** will present the contents of the current working directory
- Filenames starting with a dot (.) are normally hidden
- The command **ls** can be used to list many additional properties of individual files, including size and permissions.

Exercise 1

Find out how to use the **ls** command to list the contents of your home directory, sorted in descending order based on file creation time. Use **ls --help** and **man ls** to find more information about the many options of the **ls** command. Which of these two provides more insights do you think?

Standard input and output

Every program uses three data streams:

- Standard input
  - This will normally be your keyboard
- Standard output
  - This will normally your display
- Standard error
  - Information about things that go wrong (less relevant for now).

These data streams can be assigned in different ways and this is an incredibly useful feature.

A chain of commands

The pipe symbol (vertical bar, '|') can be used to assign the standard output of the first command as the standard input of the second command.

Example: **ls** | **sort** first executes the command **ls**, but instead of displaying its output on the screen, it gives its output as input to the command **sort**.

To and from files

- A greater-than symbol ('>') writes the standard output to a file
  - Example: **ls** > **myfiles.txt** stores the output of the command **ls** in the file **myfiles.txt**.
- A less-than symbol ('<') uses the contents of a file as standard input
  - Example: **sort** < **mydata.csv** gives the contents of the file **mydata.csv** as input to the **sort** command.

Exercise 2

Execute the following chain of commands and observe the results:

printf "a\nb\n" | sort -r

Explain what happens here.

Other commands often used in Bash

Some other handy utilities:

- Command line calculator **bc**
- The **file** tool to detect file types
- Concatenate a file to standard output using **cat**
- Print the current working directory by invoking **pwd**
- Copy files (**cp** command), moving files (**mv** command) and removing files (**rm** command)

Part II: Text and data

Creating your own text files

Several well-known editors:

- **vim**
- **nano**
- **emacs**

Others are available as well. Please note that it can take some time to become familiar with editing a text-file in such a way.

Advanced text search and text manipulation

We will be mostly concerned with **grep** and **sed**

- **grep** (Get Regular Expression Pattern) matches lines in a file based on a pattern
- **sed** (Stream EDitor) modifies lines in a file and can therefore be very effectively used within a chain of commands

Both **grep** and **sed** use regular expressions

Regular Expressions

- A regular expression is a formal description of a set of pieces of text.
- We say that a piece of text (also known as a string) is matched by a regular expression if the string is described by the regular expression.

Examples of regular expressions

- **a*** matches **a**, **aaa** and **aaaaaa**, but also the empty string
- A dot ('**.**') matches any single character
- Square brackets can be used to specify a range of characters, for example: **[a-h]*** matches: **aha** and **bee**.
- Escaped parentheses can be used to surround a regular expression and refer back to it, for example: **\(h.*o\).*\1** matches: **hello my friends, hello**.
- A caret ('^') matches the beginning of a line and the dollar ('**$**') matches the end of a line.

Using regular expressions in grep and sed

- The command **grep** outputs only those lines that match a given regular expression, and therefore can be used to effectively select relevant lines from a file, or standard input.
- The command **sed** replaces every line that matches a given regular expression with another expression.

Exercise 3

Assume that you have been given a list of Dutch names (first name, followed by last name) as lines in a file.

- Use **grep** to match every line where the last name starts with 'Jan'.
- Use **sed** to convert every first name into an initial followed by a dot.
- Combine these two in a chain of commands in bash.

Exercise 4 (difficult!)

Use the tool **wget** to retrieve the contents of a web page of your own choice that you often visit. Subsequently, use the tools **grep** and **sed** to build a chain of commands in bash to filter out precisely the information that is relevant for you at this particular website.

- Hint: Use the tool **sed** to replace HTML-tags with empty strings.