HPC.NRW

INTRODUCTION TO LINUX

(in an HPC context)

Version 20.09 | HPC.NRW Competence Network

THE COMPETENCE NETWORK FOR HIGH-PERFORMANCE COMPUTING IN NRW.

TEXT DISPLAY AND SEARCH

HPC.NRW Competence Network

INTRODUCTION TO LINUX

CONSOLE INPUT AND OUTPUT

- Console has three main ways of communicating with process (so-called streams)
  - Standard input (`stdin`)
  - Standard output (`stdout`)
  - Standard error (`stderr`)

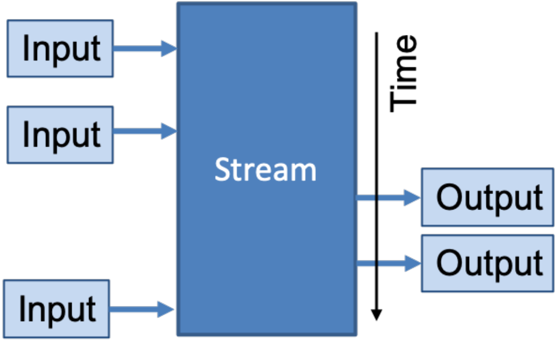
- `stdin`: what you type into console

- `stdout` + `stderr`: what you see in console
  - Two separate streams so you can separate error messages from normal output

WHY THE TERM "STREAM"

HPC.NRW

- What is a “stream” in computing terms?
  - Intermediate storage
  - Input and output may overlap
- Example: streaming video
  - Video gets partially downloaded, you can already view it
- In console: text gets written into stream and taken out
  - Input and output can be (re)directed to other sources/targets



CONSOLE INPUT AND OUTPUT

- Input/output streams can be redirected
  - Other commands
  - Files
- Redirect stdout

```
command > filename
```

- Redirect stderr

```
command 2> filename
```

- Redirect stdin

```
command < filename
```

- Use output of one command as input to another: pipe symbol

```
command1 | command2
```

ADVANCED REDIRECTION

- Stream redirection can do even more

- `command >> filename` append to file without overwriting

- Streams are numbered:

  0: `stdin`, **1**: `stdout`, **2**: `stderr`

  - Examples:

    ```
    command > out.log 2> err.log
    ```

    ```
    command 2>&1 > out_err.log
    ```

TEXT DISPLAY

- Many different ways to display and edit text
  - Simplest: `cat` command
    - Outputs contents of a text file to console
  - More advanced: `less` command
    - Allows going back and forth
    - Also used by man pages
  - Others:
    - `head`: display first lines
    - `tail`: display last lines

SEARCHING FILE CONTENTS

- Use `grep` command
- Syntax: `grep <options> <string> <filename>`
  - Example `grep -i -r "test" example*.txt`
- Like `find`, very powerful due to options + wildcards
- Common options:
  - `-r` Recursive (include subdirectories)
  - `-i` Ignore upper/lower case
  - `-I` Ignore binary files (capital i)

USE GREP ON SEARCH RESULTS

- Common situation:
  - Command with a lot of text output
  - You are looking for something inside output
- Solution: pipe output into `grep`

  ```
  $ ll | grep -i test
  ```

- Note that there is no file specified in the `grep` call
- See how pipes can be useful?