Bash command structure

Commands

- commands can be one of four things
  1. a shell built-in command (e.g., **`cd`**, **`pwd`**)
  2. an executable program
     1. compiled (e.g., C)
     2. script (e.g., shell scripting language, Perl, Python)
  3. aliases
     - a keyboard shortcut, an abbreviation, a means of avoiding typing a long command sequence
     - user-defined commands built from other commands
  4. shell functions
     - function written in the shell scripting language

- The **`type`** command displays information about the command type

```
type ls
type cd
type la
```

- The **`which`** command displays the pathname of an executable command

```
which ls
which cd
which la
```

The **`ls`** command lists information about the current directory by default, but more generally, it lists information about files:

```
cd /bin
ls ls
```

The last example was not very informative

- let's try to find out how large the **`ls`** program is

```
ls -s ls
```

By default, the size is given in units called blocks

- to get the size in KB, MB, GB, … use:

```
ls -sh ls
```

Command structure

- shell commands in Linux have the following structure

  `command` *`-options arguments`*

  where:

  - **`command`** is the name of the command,
  - *`options`* are optional flags that modify the behavior of the command,
  - *`arguments`* are the space-separated list of items on which the command acts

Options can be combined

- one hyphen, no spaces between options

```
ls -Ash1
```

Options can be specified separately

- one hyphen per option, space between options

```
ls -A -s -h -1
```

Long-form options exist for some options

- better readability but longer to type in
- two hypens before each option, space between options

```
ls --almost-all --human-readable --size -1
```

Commonly used `ls` options

Option	Long option	Description
`-a`	`--all`	List all files (even hidden)
`-A`	`--almost-all`	Like `-a` but does not list `.` or `..`
`-d`	`--directory`	List the directory name instead of its contents
`-F`	`--classify`	Appends an additional character to indicate file type
`-h`	`--human-readable`	With `-l` and `-s` specify size in human readable format
`-l`		Long format
`-r`	`--reverse`	Reverse order
`-s`		Print file size in blocks
`-t`		Sort by modification time
`-1`		One file per line

Many commands will accept the **`--help`** option to display some documentation for the command

```
ls --help
cd --help
```

For shell built-in commands, use **`man builtins`** or the **`help`** command

```
man builtins
help cd
```

Executable commands intended for command-line usage often have formal documentation in the form of a manual or man page

- intended as reference, not tutorial

```
man ls
```

In the SYNOPSIS, anything in square brackets is optional

- … means multiple items are allowed

```
LS(1)                                                   User Commands
  LS(1)

NAME
       ls - list directory contents

SYNOPSIS
       ls [OPTION]... [FILE]...

DESCRIPTION
       List  information  about the FILEs (the current directory by
  default).  Sort entries alphabetically if none of
       -cftuvSUX nor --sort is specified.
```

Navigating man pages

- displayed using the **`less`** program
- some useful navigation keys are listed below
  - **`^[key]`** means **`CTRL-[key]`**

Key	Description
`e, ^E, j, ^N, ENTER`	Forward one line (or N lines)
`y, ^Y, k, ^K, ^P`	Backward one line (or N lines)
`f, ^F, ^V, SPACE`	Forward one window (or N lines)
`b, ^B, ESC-v`	Backward one window (or N lines)
`z`	Forward one window (and set window to N lines)
`w`	Backward one window (and set window to N lines)
`d`	Forward one half-window (and set half-window to N lines)
`u`	Backward one half-window (and set half-window to N lines)

The **`cowsay`** program has some fun options:

```
man cowsay
```

You can use the up and down arrows on your keyboard to navigate through the history of commands

- you can use the **`history`** command to view the history of commands
- use **`!`*`num`*** to repeat a command from the history

```
history
```

The Bash shell has a feature called tab completion

- start typing a command and press the TAB key
- will also complete filename arguments to commands

```
cow<press TAB here>
ls /usr/games/cow<press TAB here>
ls /usr/games/p<press TAB here>
```

man page sections

- sometimes need to specify which section of the manuals to search
  - occurs when there are multiple commands with same name

Section	Description
1	General commands
2	System calls
3	Library functions (C standard library)
4	Special files (usually devices and drivers)
5	File formats and conventions
6	Games and screensavers
7	Miscellaneous
8	System administration commands

Wildcards

- filenames /pathnames are used very often by the shell
- the shell provides special characters called *wildcards* to help succinctly specify groups of filenames
  - also called *globbing patterns* (**`man 7 glob`**)

Wildcard	Description
`*`	Matches any number of any characters
`?`	Matches any single character
`[`*`characters`*`]`	Matches any character in the set ***`characters`***
`[!`*`characters`*`]`	Matches any character not in the set ***`characters`***
`[[:class:]]`	Matches any character in the specified class

Wildcard examples

Pattern	Matches
`*`	All files
`a*`	All files starting with `a`
`*.txt`	All files ending with `.txt`
`a*.txt`	All files starting with `a` and ending with `.txt`
`???`	Any three character filename
`x?z`	Any three character filename starting with `x` and ending with `z`
`x[yY12]z`	`xyz` or `xYz` or `x1z` or `x2z`
`x[a-z]z`	Any three character filename starting with `x`, followed by a lowercase letter between `a` and `z`, and ending with `z`
`x[0-9]z`	Any three character filename starting with `x`, followed by a digit, and ending with `z`
`[0-9][0-9].pdf`	Any filename starting with two digits and ending in `.pdf`

Character classes

- some common character classes

Class	Meaning
`[:alnum:]`	Alphanumeric characters
`[:alpha:]`	Alphabetic characters
`[:digit:]`	Digits
`[:lower:]`	Lowercase letters
`[:upper:]`	Uppercase letters

Wildcard class examples

Pattern	Matches
`[[:upper:]]*`	All files starting with **A** through **Z**
`[![:upper:]]*`	All files not starting with **A** through **Z**
`x[[:digit:]]`	Any two character filename starting with **x** and ending with a digit