Shell commands cheat sheet

The following is a summary of UNIX shell commands used in this workshop. They are listed in alphabetical(-ish) order. These are just a few of the many, many UNIX commands that exist.

cat

> **COMMAND:** cat
> Reads the content of the file
> **DEFAULT BEHAVIOR:** Directs output to your screen##grep

grep

> **COMMAND:** grep
> Identifies lines within your file that match a pattern
> **FLAG:** -i
> Enforces case-insensitivity
> **FLAG:** -v
> Inverts the results to output only the lines that *don't* match the pattern
> **FLAG:** -E
> Tells the program that we are going to be using a regular expression for the pattern that it needs to match

rm

> **COMMAND:** rm
> Removes an entry from a directory

##sort

> **COMMAND:** sort
> Sorts the lines of a text file
> **DEFAULT BEHAVIOR:** Writes the results to your screen
> **FLAG:** -o
> Redirects output to a file.
> **WARNING:** This is EXTREMELY slow, so it's better to redirect the output into another command if you can.

uniq

> **COMMAND:** uniq
> Reports or filters out repeated adjacent lines in a file
> **DEFAULT BEHAVIOR:** Outputs one copy of each line in the file
> **FLAG:** -u
> Outputs each unique line in the file

other stuff

| (pipe)

> **COMMAND:** |
> Redirects the output from one command directly into the next command as input##>

>

> **COMMAND:** >
> Redirects content into a file instead of to your screen
> **WARNING:** Overwrites the content of an existing file without warning!

>>

> **COMMAND:** >>
> Appends the redirected content to an existing file, instead of overwriting it###built-in variables

built-in variables

> **VARIABLES:** $1 and $2
> These are special variables (actually $1 - $9) that are built into the UNIX system and correspond to the arguments passed to the program when it is run.

running a script

> **COMMAND:** bash [filename.sh]

shebang

Tells the computer which interpreter to use to run the script. The shebang below indicates we are using the bash shell.

```
#!/bin/bash
```