OSU Center for Genome Research and Biocomputing (CGRB), MCB 599: "Introduction to Unix/Linux" (INX_U18)
"Cheat Sheet": Day 02, **Pages:** 3, **Revision:** 2018-07-23, **Instructor:** Matthew Peterson (matthew@cgrb.oregonstate.edu)

Command/Concept	Usage/Syntax	Description	Example
`echo`	`echo <some string>`	Displays a line of text on the terminal.	`echo hello`

Environment variables	Built-in variables and ones you set start with a `$`	Bits of information (preferences) stored in your shell. Can be used by you or by other programs/scripts.	# Your username
			`echo $USER`
			# Program you are running
			# In this case your shell
			`echo $0`

`env`	`env`	List all of the environment variables currently set in your shell. **Note:** This command is tcsh specific.	`env`

`bash`	`bash`	Starts the bash shell (by default we are running tcsh). Type `exit` to quit out of the bash shell.	# Start a bash shell
			`bash`
			# Show the shell running
			`echo $0`
			# Exit the shell
			`exit`

`setenv`	`setenv <varname>`
	`<value>`
		Set an environment variable. **Note:** This command is tcsh specific.	# Set variable greeting
			`setenv greeting hello`
			# See value of greeting
			`echo $greeting`

`'` (Single quotes)	`'<some string>'`	Surround a string with single quotes such that it is treating it as a single value, parameter, or argument. Needed to "escape" spaces in strings. Single quotes <u>do not</u> expand the content of variables inside them.	# Not using quotes in this
			# example returns an error
			`setenv greeting`
			`'hello class'`
			# Variable <u>not</u> expanded
			# greeting = hello $USER
			`setenv greeting`
			`'hello $USER'`

" (Double quotes)	"<some string>"	Surround a string with double quotes such that it is treating it as a single value, parameter, or argument. Needed to "escape" spaces in strings. Double quotes do expand the content of variables inside them.	# Not using quotes in this # example returns an error setenv greeting "hello class" # Variable is expanded # greeting = # hello <your username> setenv greeting "hello $USER"
\<space>	\<space>	Command line arguments are usually separated by spaces. Spaces can be "escaped" such that a string is treated as a single value, parameter, or argument (without quotes).	# Not using \<space> in this # example returns an error setenv greeting hello\ class
pwd	pwd	Print Working Directory. Also available as an environment variable Present Working Directory: $PWD	# Command pwd # Environment variable echo $PWD
cd	cd <directory name>	Change (working) directory	cd / cd /<*dir1*>/<*dir2*>
Options for going back to your home directory	cd <absolute path to your $HOME dir> cd $HOME cd cd ~	Four options to go back to your home directory no matter what your $PWD is.	cd <absolute path to your $HOME dir> cd $HOME cd cd ~
ls	ls	List files and directories in your $PWD. Running ls after each cd is a good way to familiarize yourself with the filesystem.	ls

`ls -a`	`ls -a`	List all files and directories in your `$PWD` including "hidden dot files," they start with either . or ..	`ls –a`
`ls -l`	`ls -l`	List files and directories in your `$PWD` using "long" format. Provides permission, size, and other properties of each file and directory.	`ls –l`
`ls -l -h`	`ls -l -h`	List files and directories in your `$PWD` using "long" format. File sizes are listed as "human readable" (instead as just bytes), i.e., **Ki**lobytes, **M**egabytes, **G**igabytes, and **T**erabytes.	`ls -l –h`

Consolidation of arguments	`ls -<multiple args>`	Order does not matter. Not all commands support "merging" of multiple options.	`ls -l -h -a`
			`ls -lha`
			`ls -hal`
			`ls –lah`