Bash, Find, Grep and Regular Expression Cheat Sheet

bash

- **Variables**:
```
var=value              # No spaces around '='.
```
Access: `$var`, e.g., `echo $var`.

- **Arithmetic**:

let `var=expression` or `var=$((expression))`

Example: `sum=$(($1 + $2))`

- **Control Structures**:

If-Else:
```
 if [ condition ]; then
      # code
  elif [ condition ]; then
      # code
  else
      # code
  fi
```
- **For Loop**:
```
for var in list; do
    # code
done
```
- **Functions**:
```
my_function() {
    # code
  }
```
Call function: `my_function`

Bash parameter expansion:

Uppercase Conversion
```
${var^^}         # Converts entire string to uppercase.
```

Lowercase Conversion
```
${var,,}        #  Converts entire string to lowercase.
```

Suffix Removal

- Smallest matching suffix: `${var%pattern}`
- Largest matching suffix: `${var%%pattern}`
```
filename="/home/user/file.txt"
echo "${filename%/*}"  # Output: /home/user
```

Prefix Removal

- Smallest matching prefix: `${var#pattern}`
- Largest matching prefix: `${var##pattern}`
```
filename="home/user/file.txt"
echo "${filename##*/}" # Output: file.txt
```

find

- **Basic Syntax**:

find [directory] [criteria] [action]

- **Common Criteria**:
  - Name: -name "*.txt" → Find files by name.
  - Type: -type f → Find files, -type d → Find directories.
  - Size: -size +100k → Files larger than 100KB.
  - -mtime +n → Modified more than n days ago.
  - -atime -n → Accessed less than n days ago.
  - -or or -o → Either condition can match when combining multiple conditions
- **Action**:
  - Execute Command: -exec command {} \;

Example: Delete all log files in current directory
```
find . -name "*.log" -exec rm {} \;
```

grep

- **Basic Syntax:**

grep [options] pattern [file(s)]

- **Common Options**:
  - -r → Recursive (search directories).
  - -w → Match whole words.
  - -v → Invert match (show lines not matching pattern).
  - -n → Show line numbers.
  - -l → Only list filenames that match.
- **Examples**:
  - Find lines containing "error" in all `.log` files:
    `grep "error" *.log`
  - Search recursively for "main" in all `.c` files:
    `grep -r "main" *.c`
  - Count occurrences of "TODO" in a file:
    `grep -c "TODO" file.txt`

Basic regular expressions (default in sed and grep):

.	matches any character including newline
^	matches beginning of the line
$	matches end of the line
*	matches zero or more of the preceding character, group or bracketed list
[*list*]	matches one character to any of the characters **listed** within the brackets. Range abbreviations can be used (e.g., [a-f], [0-9])
[^*list*]	matches one character if it is **not listed** within the brackets.

Extended regular expressions (requires "-r" in sed or "-E" in grep to use natively, without needing escape characters):

\(*regexp*\)	groups the inner *regexp* and allows it to be referenced later with \1, \2, etc
\+	similar to *, but matches one or more
\{*i*\}	matches *i* occurrences of the preceding character, group or bracketed list
\{*i*,\}	matches *i* or more occurrences of the preceding character, group or bracketed list
\{*i*,*j*\}	matches between *i* and *j* occurrences of the preceding character, group or bracketed list
\?	matches zero or one instance of the preceding; equivalent to \{0,1\}
pat1\\|*pat2*	alternation: matches either *pat1* or *pat2*