Unix Cheat Sheet

AWK

awk [options] 'function func(x){return x} BEGIN {before file is read} regex/condition {if matches} {always} END {after file is read}'
options: -F [field separator] -v [x=val]
functions: substr(string, start, length), getline, x%y==z, next, int()
variables: FILENAME, FS, OFS(for output), RS(line separator), ORS, NF, NR, FNR(for each file)

```
eg awk '{x[NR]=$0; s+=$0;} END{a=s/NR; for (i in x){ss += (x[i]-a)^2} sd = sqrt(ss/NR); print "Mean = "a" SD = "sd}'
eg awk '{arr[$2]=arr[$2]" "$1}END{for (a in arr) print a, arr[a]}'
```

CAT

cat -A(show whitespace delimiters) -b(number nonblank lines) -n(number lines)

```
eg head input | cat -A
```

CUT

cut [-b[bytes]|-c[chars]|-f[fields]] --complement
ranges: N, N-, N-M, -M

```
eg cut -f4- input
```

FIND

find [-P(don't follow symbolic links)|-L(follow links)] [path] [expression[options tests actions]]
options: -maxdepth [levels below pathdir] -mindepth [levels below pathdir] -mout(this fs only) -regextype [posix-awk|posix-basic|posix-egrep|posix-extended]
tests: [+n(>n)|-n(<n)|n(=n)] -empty --mmin [# minutes ago file was mod -mtime [# days ago file was mod] -name [pattern] -newer [file] -path [pattern] -regex [pattern] -size [bytes][c(bytes)|k|M|G] -wholename [pattern] *all [pattern] options have -i[option] versions*
actions: -delete [-print|-fprint file(print filename)] -exec[command|dir command(run inside dir where file was found)| command {} + (copies filenames into one command line)]

```
eg find dmel*dpse*/* -exec cp {} ~/dros/ \;
```

GREP

grep [options] [pattern]
options: -A(after) -B(before) -C(before & after) --color [-H(with filename)|-h(no filename)] -i(case insensitive) -n(prefix with line number) -o(only show part that matches) -r(recursive) [--include=[filename pattern]|--exclude="filename pattern" (use with -r)] -v(inverse)
different output: -c(count of matches) [-l(files without matches)|-L(files with matches]
regex: [-E(extended)|-P(perl)] -w(makes entire word) -x(makes entire line)

```
eg grep -A 5 -n "ENSMUSP00000080"  input
```

HEAD

head --bytes=[-N] --lines=[-N]

```
eg head -442
```

JOIN

join [options] file1 file2
input: -i(case insensitive) -1(join on this field) -2(join on this field)
output: -a(print unpairable lines)[filenum] -e[replace missing fields with this] -o[0(match) 1.* 2.*] -t[tab delimiter(Ctrl+V, then type char)] -v(print only unpaired lines)

```
eg join -a1 -a2 -e "BON" -o 0 1.1 2.1 mouse.a mouse.f > mouse.join | grep "BON"
```

LS
ls -a(incl. hidden) --author -d(don't show directory contents) -F(classify file types) -h(sizes formatted) -l(long) -r(reverse) -R(list subdirectories) -S(sort by file size) -t(sort by mod time)
```
eg ls -lth
```

PASTE
paste -d[tab delimiter] file1 file2

PERL
perl -c(compile only) -e(command line program) -i(edit in place) -l[record separator] -n(put in while(<>) loop) 'code;'
```
eg perl -n -e '$on=(/^>.+f.(\S+)[.].*\t(\S*)/); print $1."\t".$2."\n";' dmel_oneline.fasta
```

SED
sed –r 's/replace_this/with_this/g'
replace_this: [^^][^^]*(anything not a space), \("match me"\)
with_this: &=matched chars, \1=only print matched chars, /1=print only first occurrence

SORT
sort [options] input
what to sort: -b(ignore leading blanks) -d(consider only blanks & alphanum) -f(case insensitive) -g(numbers containing sci. notation) -n(numbers)
how to sort: sort -c(check if sorted) -k[start field],[end field] -m(merge, don't sort) -t $'[tab delimiter (always use with -k)], -r(reverse)
```
eg sort -n -k3,3 -t $'\t' input
```

SPLIT
split [options] [input] [prefix]
options: -a[suffix length] -b[bytes per file] -d(numeric suffixes) -l[lines per file]
```
eg split -l 2670 -d scer.final scer.final
```

TR
tr -d [-c(complement)] "set to remove" < input OR tr "replace this" "with this" < input
reads from STDIN ONLY
sets: char1-char2 (range), [:SET:], where SET can = alnum (alpha + num), alpha, blank (whitespace), digit, lower, upper, punct
```
eg tr -d "\15\32" input
```

UNIQ
uniq -c(count) -d(output only duplicate lines) -f[skip this # of fields] -i(case insensitive) -s[skip this # of chars] -u(output only unique lines) -w[compare this # of chars]
```
eg uniq -c input
```

WC
wc -b(bytes) -l(\n counts) -L(length of longest line) -m(chars) -w(words)
```
eg wc -l *mutants*
```