Linux Short Cheatsheet

```
wc -l ①
```

① Zählen von Zeilen

```
cut -d'|' -f1,3 file.txt ① ②
```

① Gibt als Trennzeichen eine Pipe an

② Es werden nur die erste und dritte Spalte ausgegeben

```
sort -t'|' -k1 -r -n file <1> <2> ③ ④
```

① Als Trennzeichen wird eine Pipe angegeben

② Es wird nach der ersten Spalte sortiert

③ In umgekehrter Reihenfolge sortieren

④ Nummerisch sortieren

```
uniq -c file ①
```

① Lösche doppelte Werte und gibt die Anzahl der gleich vorkommenden Werte aus

```
grep -Evc 'H[ae]llo' test.txt <1> ② ③
```

① Mit `-c` werden die gefundenen Zeilen gezählt

② Um Extended-Regex nutzen zu können muss die Option `-E` gegeben sein

③ Gibt alle nicht gematched Zeilen zurück

Extended Regex von `grep` ist unterschiedlich zum normalen Regex.

- `[[:alnum:]]` Alphanumerische Characters
- `[[:alpha:]]` Alphabetische Characters
- `[[:blank:]]` Space und Tab
- `[[:digit:]]` Ziffern

- `[[:lower:]]` Kleinschreibung
- `[[:space:]]` Tab, Newline, Vertical Tab, Page Break, Carriage Return, und Space
- `[[:upper:]]` Großschreibung

```
sed 's/<regex>/<replace>/' file ①
sed -r 's/<regex>/<replace>/<flag>' file ②
sed 'n,ms/<pattern>/replace/' ③
sed '/<line_pattern>/s/<find>/<replace>/' <file> ④
sed '/<regex>/d' <file> ⑤
sed -n '<n>,/^$/p' <file> ⑥
sed 's#<regex>#<replace>#' <file> ⑦
sed -r 'n,m {<commands>}' ⑧
```

① Ersetzt pro Zeile einmal den `<regex>` durch `<replace>`

② Mit `-r` wird der extended Regex genutzt. Am Ende kann auch eine Flag angegeben werden, z.B. die `g` Global Flag, mit dieser wird nicht pro Zeile, sondern alles gematched

③ Ersetze das pattern in den Zeilen `n` - `m`

④ Ersetzt nur die Matches in den Zeilen, welche das `<line_pattern>` erfüllen

⑤ Löscht die Zeilen, welche einen Match haben

⑥ Gibt alle Zeilen von `<n>` bis zur nächsten leeren Zeile aus

⑦ Ein Slash `/` als Trennzeichen kann auch durch ein anderes ersetzt werden, z.B. eine Raute `#`

⑧ Führe `command` beim `n` ten auftreten bis Zeile `m` aus

```
for i in * ①
IFS=';' ②
for el in $var ②
while read v1 v2 v3; do ③
done < $file ③
if [ ] && [ ] ④
```

① Gehe alle Files im momentanen Directory durch

② Gehe alle Elemente durch welche mit `IFS` separiert sind

③ Lese zeilenweise ein File aus, die Variablen bei read werden je nach `IFS` eingefügt

④ If Verzweigung

Zahlenwert testen	
`-eq`	equals
`-ne`	not equals
`-lt`	less than
`-le`	less or equal than
`-gt`	greater than
`-ge`	greater or equals than
Files/Directories Testen	
`-d`	Ist ein Verzeichnis
`-e`	File existiert
`-f`	Reguläre Datei
`-r`	Benutzer Leserechte auf File
`-w`	Benutzer Schreibrechte auf File
`-x`	Benutzer Ausführrechte auf File
`-s`	Datei nicht leer
Dateivergleich	
`-nt`	Datei1 neuer als Datei2
`-ot`	Datei1 älter als Datei2
`-ef`	Dateien selbe Inode-Nummer (Hardlinks)
Zeichenkettenvergleich	
`=`	Zeichenketten gleich
`!=`	Zeichenketten unterschiedlich
`-z`	Zeichenkette leer
`-n`	Zeichenkette nicht leer
Logische Verknüpfungen	
`!`	Not
`-a`	And

`-o`	Or

Table 1. Table Vergleichsoperatoren