Notes on grep

Basic Regular Expressions (BREs)

Single character matching

- `b` matches b, `A` matches A etc.
- metacharacters are searched for by escaping them: `\[` matches [, `\$` matches $.
- dot: `.` matches any single character. `a.c` –> aac, abc, aqc etc.
- bracket: `c[aeio]t` –> cat, cet etc.
  - `[^<stuff>]` –> match any pattern NOT with stuff
  - metacharacters within brackets aren't special and stand for their literal meanings. To get `]` or `-` into the set, put it first in the list. To get both, put `]` in the start and `-` in the end.
  - range: `[1-9]` –> match any of 1-9
- (**optional**) Collating elements and equivalence classes: depending on the locale, [[=e=]] (an equivalence class) –> e, é, ë etc. Also [.ch.] (a collating element) treats `ch` as a single unit for sorting and stuff. It WILL NOT match c/h.
- character classes: [[:alpha:]] –> single alphabets, [[:digit:]] –> single digits. Others include [:punct:] for punctuation, [:lower/upper:] for lower/uppercase characters, [:alnum:] for alphanumeric characters etc.

Backreferences

1. enclose pattern you want to reference in (and).
2. call a backreference using `\digit` where digit = 1-9.

Eg: `\(why\).*\1` –> any sentence with a `why` in front and at the end.

Modifiers aka multiple character matching

- asterisk: matches zero or more of the single preceding character. `ab*c` –> ac, abc, abbbbc etc.
- interval expressions:
  - {n} –> find *exactly* n occurences of preceding RE
  - {n, } –> find *at least* n occurences of preceding RE
  - {n, m} –> find between n and m occurences of preceding RE.

Note: m <= `RE_DUP_MAX` (get using `getconf RE_DUP_MAX`)

Anchors

- carat: search for RE at the start of string.
- dollar: search for RE at the end of string.

Note: `^$` literally means look for lines only with *zero-width strings* aka empty lines. Use together with the invert option (`-v`) to remove empty lines.

Operator Precedence

1. `[..]`, `[==]`,`[::]` (bracket collation)
2. `\metacharacter`
3. `[]` (bracket expressions)
4. `\(\) \digit` (subexpressions & backreferences)
5. `*`, `\{\}` (repetition of preceding RE)
6. concatenation
7. `^`, `$` (anchors)

GNU specific (special) regexs

- `\w`: matches word-constituent (a word is basically letters+digits+underscores) character. Equivalent to `[[:alnum:]_]`.
- `\W`: matches nonword-constituent characters. Equivalent to `[^[:alnum:]_]`.
- `\< \>`: matches beginning and end of a word. Eg: `\<chop` matches "chopstick" but not "eat a lambchop".