Regular Expressions Sheet

Source: ***https://github.com/tartley/python-regex-cheatsheet/blob/master/cheatsheet.rst***

Additional Source: ***https://docs.python.org/2/library/re.html***

```
\       escapes special characters.

.       matches any character

^       matches start of the string (or line if MULTILINE)

$       matches end of the string (or line if MULTILINE)

[5b-d]  matches any chars '5', 'b', 'c' or 'd'

[^a-c6] matches any char except 'a', 'b', 'c' or '6'

R|S     matches either regex R or regex S.

()      Creates a capture group, and indicates precedence.


*       0 or more   (append ? for non-greedy)

+       1 or more    "

?       0 or 1       "

{m}     exactly 'm'

{m,n}   from m to n. 'm' defaults to 0, 'n' to infinity

{m,n}?  from m to n, as few as possible

{,1}    is the same as using ?

{0,}    is the same as using *

{1,}    is the same as using +


\A  Start of string

\b  Matches empty string at word boundary (between \w and \W)

\B  Matches empty string not at word boundary

\d  Digit
```

```
\D  Non-digit

\s  Whitespace: [ \t\n\r\f\v], more if LOCALE or UNICODE

\S  Non-whitespace

\w  Alphanumeric: [0-9a-zA-Z_], or is LOCALE dependant

\W  Non-alphanumeric

\Z  End of string

\g<id>  Match previous named or numbered group,

        e.g. \g<0> or \g<name>

\a  ASCII Bell (BEL)

\f  ASCII Formfeed

\n  ASCII Linefeed

\r  ASCII Carraige return

\t  ASCII Tab

\v  ASCII Vertical tab

\\  A single backslash


\xHH   Two digit hex character

\OOO   Three digit octal char

       (or use a preceding zero, e.g. \0, \09)

\DD    Decimal number 1 to 99, matches previous

       numbered group


(?iLmsux)       Matches empty string, sets re.X flags

(?:...)         Non-capturing version of regular parentheses

(?P<name>...)   Creates a named capturing group.

(?P=name)       Matches whatever matched previously named group

(?#...)         A comment; ignored.
```

```
(?=...)         Lookahead assertion: Matches without consuming

(?!...)         Negative lookahead assertion

(?<=...)        Lookbehind assertion: Matches if preceded

(?<!...)        Negative lookbehind assertion

(?(id)yes|no)   Match 'yes' if group 'id' matched, else 'no'


re.I == re.IGNORECASE   Ignore case

re.L == re.LOCALE       Make \w, \b, and \s locale dependent

re.M == re.MULTILINE    Multiline

re.S == re.DOTALL       Dot matches all (including newline)

re.U == re.UNICODE      Make \w, \b, \d, and \s unicode dependent

re.X == re.VERBOSE      Verbose (unescaped whitespace in pattern

                        is ignored, and '#' marks comment lines)


compile(pattern[, flags]) -> RegexObject

match(pattern, string[, flags]) -> MatchObject

search(pattner, string[, flags]) -> MatchObject

findall(pattern, string[, flags]) -> list of strings

finditer(pattern, string[, flags]) -> iter of MatchObjects

split(pattern, string[, maxsplit, flags]) -> list of strings

sub(pattern, repl, string[, count, flags]) -> string

subn(pattern, repl, string[, count, flags]) -> (string, int)

escape(string) -> string

purge() # the re cache
```