Python Regular Expressions

Meta Characters (or Special Characters):

`^` → beginning of a string is a given character
(can also reference a complement of a set)

`$` → end of string is a given character

`.` → Matches any character except \n

`*` → 0 or more occurrences

`+` → 1 or more occurrences

`?` → 0 or 1 occurences
(can also be used as a non-greedy modifier)

`{n}` → exactly n occurrences

`{n,m}` → between n and m occurrences (inclusive)

`{n,}` → at least n occurrences

`|` → either-or

Functions:

import re

`re.findall(A, Z)` → finds all occurences of A in Z
returns a list of matches if found and an empty list if not

`re.search(A, Z)` → finds the first occurrence of A in Z, returns the corresponding match object if found, None if not

`re.split(A, Z)` → splits Z into a list based on A, returns the entire string as an item in the list if not found

`re.sub(A, B, Z)` → replaces all instances of A in Z into B

Match objects:

match = re.search(A, Z)

`match.span()` → where A is in Z

`match.group()` → A

`match.string()` → Z

Special Sequences:

`\d` → matches a digit character

`\D` → matches any non-digit character

`\w` → any digit, alpha, or underscore character

`\W` → non-alpha-numeric characters

`\s` → any whitespace character

`\S` → non-whitespace character

`\b` → the boundary (or empty string) at the start of the word (between \w and \W)

`\B` → where \b does not

Sets (represented w/ [ ]):

`[az]` → matches a OR z

`[a-z]` → any alpha char from a to z

`[a\-z]` → a, -, OR z

`[a-]` or `[-a]` → a OR -

`[a-z0-9]` → alphanumerical char

`[^az]` → char that is not a OR z

Note: Other special characters become literals inside a set

Groups (represented w/ ()):

`(? )` → extension notation for the character immediately following it

`(?PAZ)` → matches AZ, can be accessed w/ group name

`(?:A)` → matches A cannot be accessed after

`A(?=Z)` → matches A only if followed by B, good as a lookahead assertion and dosnt consume matches

`A(?!Z)` → matches A if it is not followed by Z (negative lookahead)

`(?<=Z)A` → matches A if Z precedes it, (positive lookback)

`(?<!Z)A` → mates A if Z does not precee it (negative lookback)

`(...){1}` → finds exactly 1 group, this can be any number

`r'\n'` → raw string prefix indicates '\n

re.I flag: allows you to ignore upper and lowercase letters

re.M flag: matches for everyline in a multiline input

What i referred to:
https://www.dataquest.io/blog/regex-cheatsheet/
https://www.debuggex.com/cheatsheet/regex/python