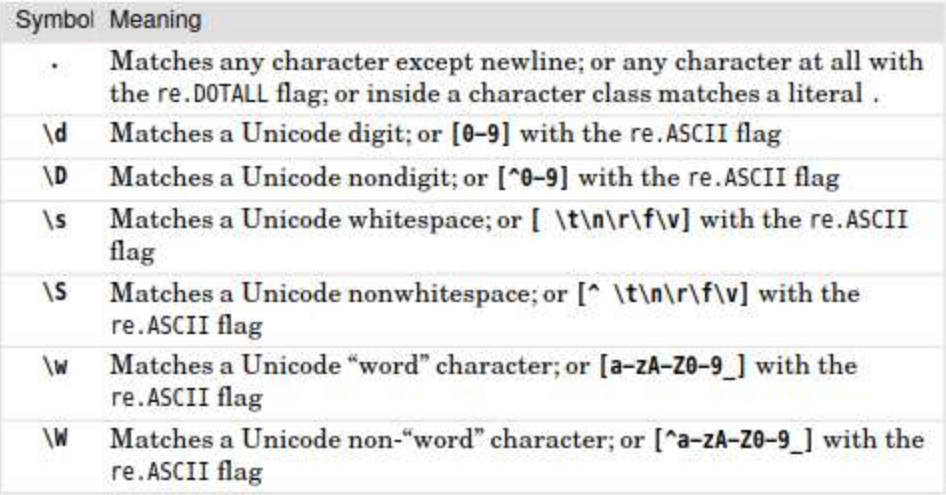
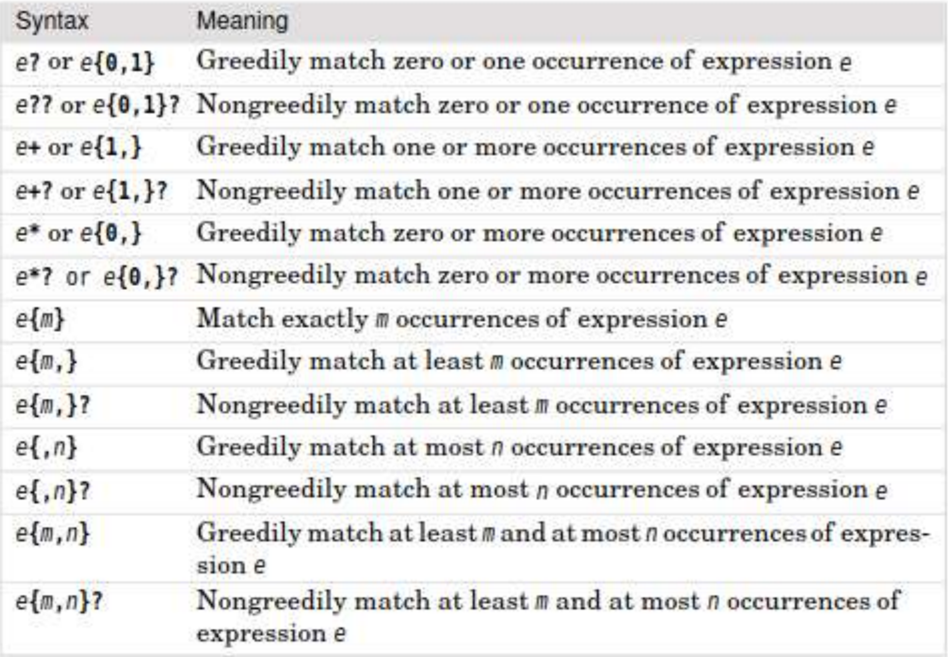


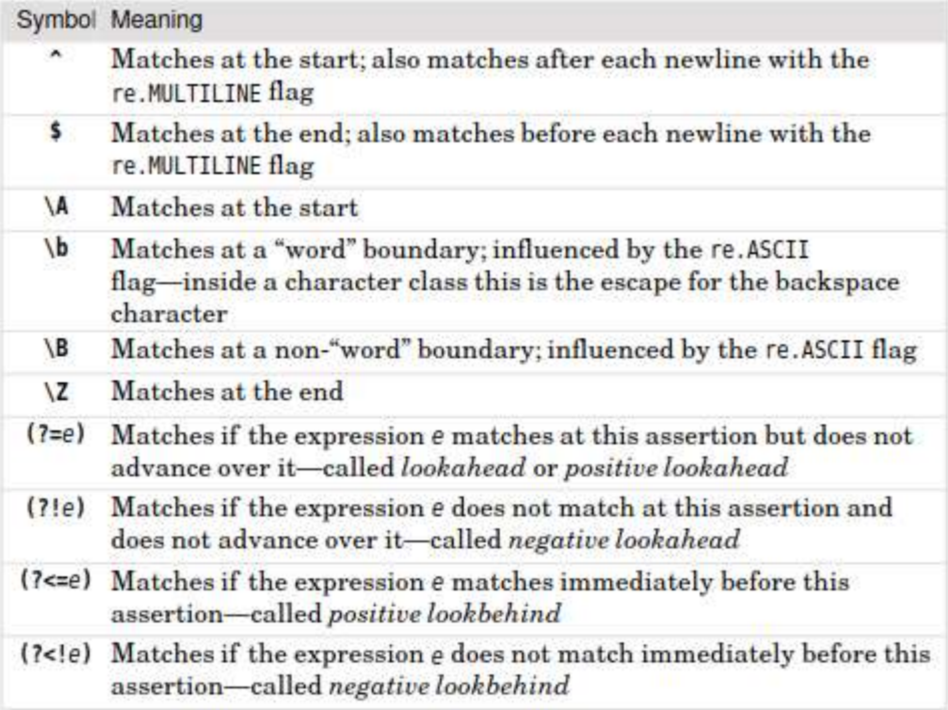
Symbol	Meaning
`.`	Matches any character except newline; or any character at all with the `re.DOTALL` flag; or inside a character class matches a literal `.`
`\d`	Matches a Unicode digit; or `[0-9]` with the `re.ASCII` flag
`\D`	Matches a Unicode nondigit; or `[^0-9]` with the `re.ASCII` flag
`\s`	Matches a Unicode whitespace; or `[ \t\n\r\f\v]` with the `re.ASCII` flag
`\S`	Matches a Unicode nonwhitespace; or `[^ \t\n\r\f\v]` with the `re.ASCII` flag
`\w`	Matches a Unicode "word" character; or `[a-zA-Z0-9_]` with the `re.ASCII` flag
`\W`	Matches a Unicode non-"word" character; or `[^a-zA-Z0-9_]` with the `re.ASCII` flag

Character Class Shorthand's



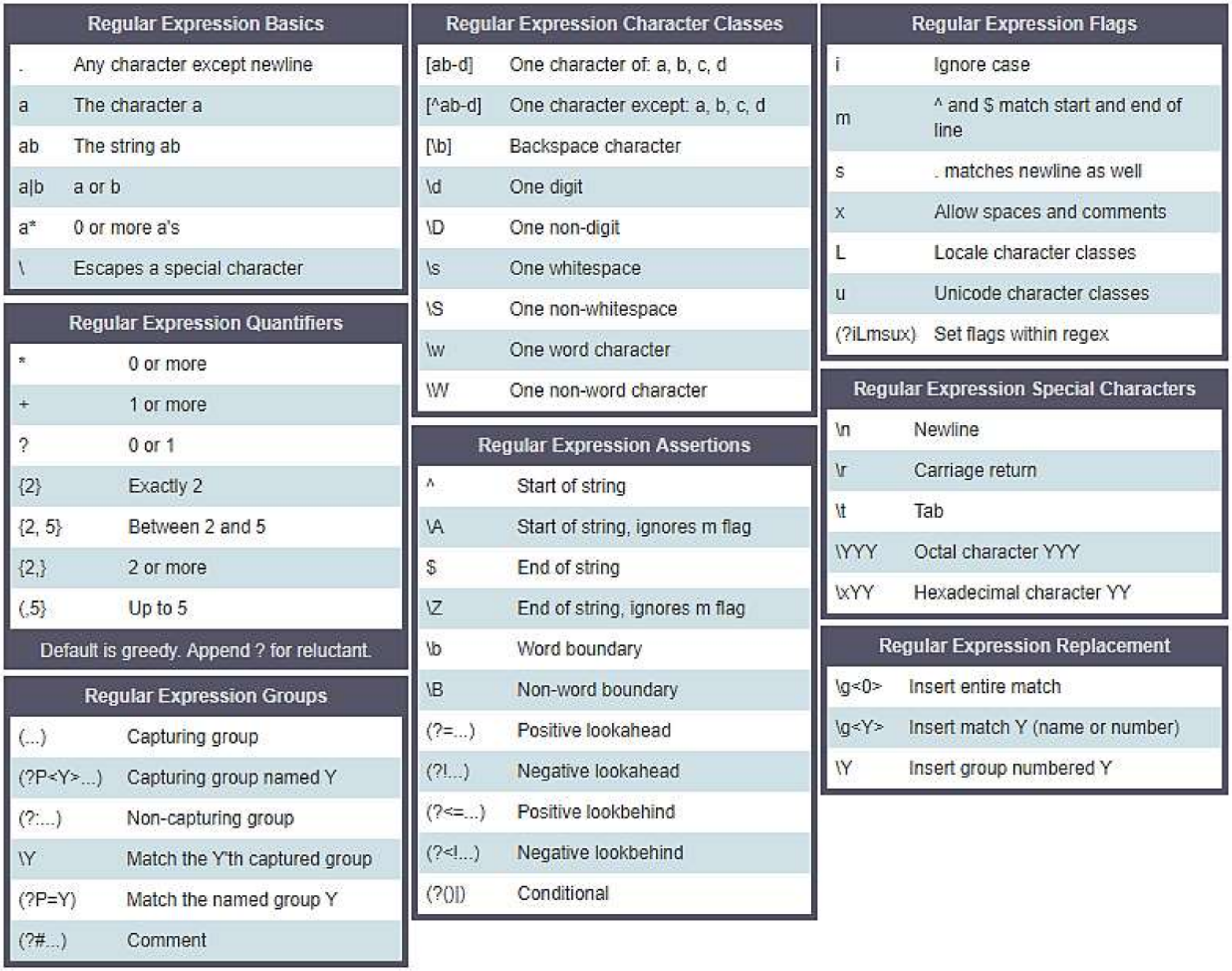
Syntax	Meaning
e`?` or *e*`{0,1}`	Greedily match zero or one occurrence of expression *e*
e`??` or *e*`{0,1}?`	Nongreedily match zero or one occurrence of expression *e*
e`+` or *e*`{1,}`	Greedily match one or more occurrences of expression *e*
e`+?` or *e*`{1,}?`	Nongreedily match one or more occurrences of expression *e*
e`*` or *e*`{0,}`	Greedily match zero or more occurrences of expression *e*
e`*?` or *e*`{0,}?`	Nongreedily match zero or more occurrences of expression *e*
e`{`*m*`}`	Match exactly *m* occurrences of expression *e*
e`{`*m*`,}`	Greedily match at least *m* occurrences of expression *e*
e`{`*m*`,}?`	Nongreedily match at least *m* occurrences of expression *e*
e`{,`*n*`}`	Greedily match at most *n* occurrences of expression *e*
e`{,`*n*`}?`	Nongreedily match at most *n* occurrences of expression *e*
e`{`*m*`,`*n*`}`	Greedily match at least *m* and at most *n* occurrences of expression *e*
e`{`*m*`,`*n*`}?`	Nongreedily match at least *m* and at most *n* occurrences of expression *e*

Regular Expression Quantifiers



Symbol	Meaning
`^`	Matches at the start; also matches after each newline with the `re.MULTILINE` flag
`$`	Matches at the end; also matches before each newline with the `re.MULTILINE` flag
`\A`	Matches at the start
`\b`	Matches at a "word" boundary; influenced by the `re.ASCII` flag—inside a character class this is the escape for the backspace character
`\B`	Matches at a non-"word" boundary; influenced by the `re.ASCII` flag
`\Z`	Matches at the end
`(?=`*e*`)`	Matches if the expression *e* matches at this assertion but does not advance over it—called *lookahead* or *positive lookahead*
`(?!`*e*`)`	Matches if the expression *e* does not match at this assertion and does not advance over it—called *negative lookahead*
`(?<=`*e*`)`	Matches if the expression *e* matches immediately before this assertion—called *positive lookbehind*
`(?<!`*e*`)`	Matches if the expression *e* does not match immediately before this assertion—called *negative lookbehind*

Regular Expression Assertions



Regular Expression Basics

.	Any character except newline
a	The character a
ab	The string ab
a|b	a or b
a*	0 or more a's
\	Escapes a special character

Regular Expression Quantifiers

*	0 or more
+	1 or more
?	0 or 1
{2}	Exactly 2
{2, 5}	Between 2 and 5
{2,}	2 or more
(,5}	Up to 5

Default is greedy. Append ? for reluctant.

Regular Expression Groups

(...)	Capturing group
(?P<Y>...)	Capturing group named Y
(?:...)	Non-capturing group
\Y	Match the Y'th captured group
(?P=Y)	Match the named group Y
(?#...)	Comment

Regular Expression Character Classes

[ab-d]	One character of: a, b, c, d
[^ab-d]	One character except: a, b, c, d
[\b]	Backspace character
\d	One digit
\D	One non-digit
\s	One whitespace
\S	One non-whitespace
\w	One word character
\W	One non-word character

Regular Expression Assertions

^	Start of string
\A	Start of string, ignores m flag
$	End of string
\Z	End of string, ignores m flag
\b	Word boundary
\B	Non-word boundary
(?=...)	Positive lookahead
(?!...)	Negative lookahead
(?<=...)	Positive lookbehind
(?<!...)	Negative lookbehind
(?()|)	Conditional

Regular Expression Flags

i	Ignore case
m	^ and $ match start and end of line
s	. matches newline as well
x	Allow spaces and comments
L	Locale character classes
u	Unicode character classes
(?iLmsux)	Set flags within regex

Regular Expression Special Characters

\n	Newline
\r	Carriage return
\t	Tab
\YYY	Octal character YYY
\xYY	Hexadecimal character YY

Regular Expression Replacement

\g<0>	Insert entire match
\g<Y>	Insert match Y (name or number)
\Y	Insert group numbered Y