Python Regular expressions with Flags

Rules to create a pattern

PART-4

Python Scripting for Automation

(Automate your repetitive tasks with python scripting)

Rules to create a pattern:

`a` , `X` , `9` - Ordinary characters that match themselves

`[abc]` - Matches `a` or `b` or `c`

`[a-c]` - Matches `a` or `b` or `c`

`[a-zA-Z0-9]` - Matches any letter from (`a` to `z`) or (`A` to `Z`) or (`0` to `9`)

`\w` - Matches any single letter, digit or underscore

`\W` - Matches any character not part of `\w`

`\d` - Matches decimal digit 0-9

`.` - Matches any single character except newline character

Note: Use \\. to match .

Rules to create a pattern:

`^` - Start of the string (and start of the line in-case of multiline string)

`$` - End of the string (and newline character in-case of multiline string)

`\b` - Empty string at the beginning or end of a word

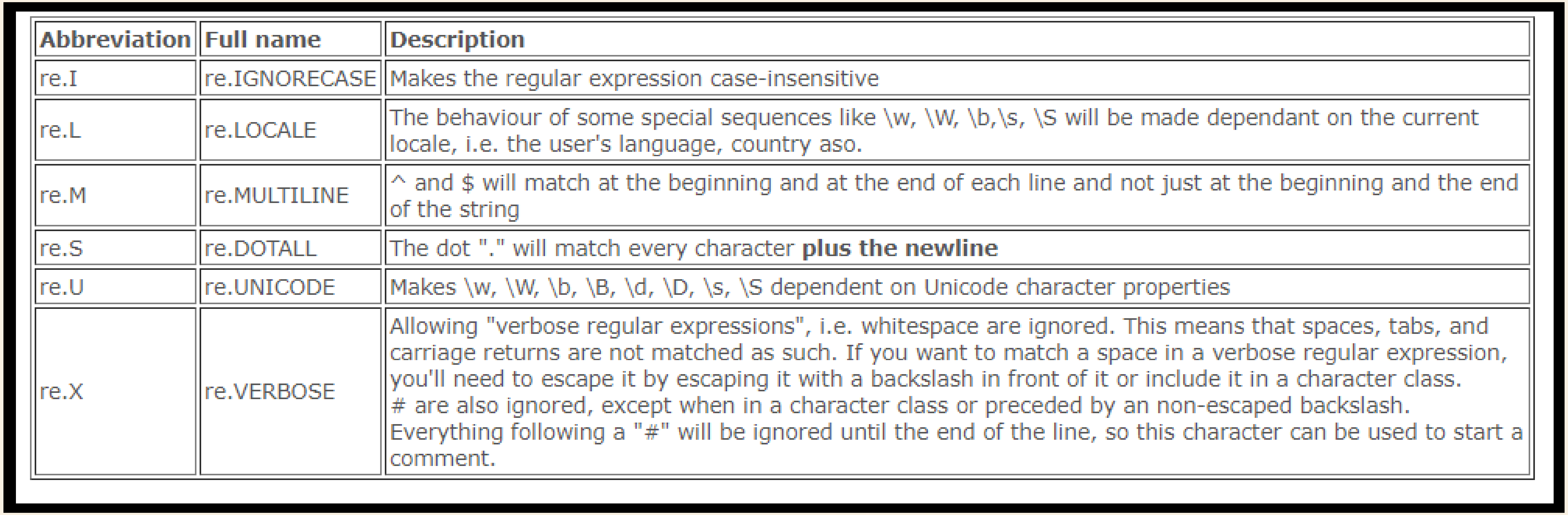
\B - Empty string not at the beginning or end of a word

`\t` , `\n` , `\r` - Matches tab, newline, return respectively

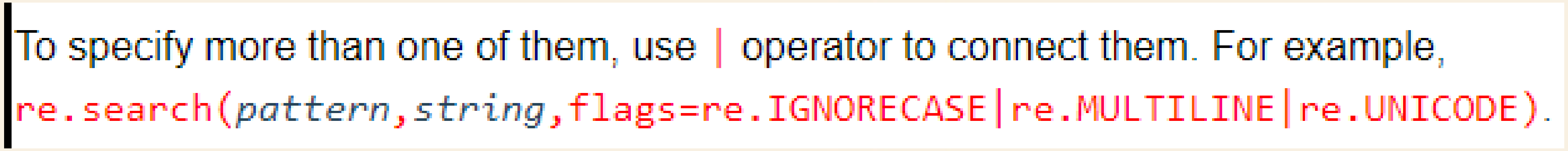
Rules to create a pattern:

{2}	exactly 2 times
{2,4}	2, 3 or 4 times
{2,}	two or more time
+	one or more
*	0 or more times
?	once or none(lazy)

Simple Flags for regex:



Abbreviation	Full name	Description
re.I	re.IGNORECASE	Makes the regular expression case-insensitive
re.L	re.LOCALE	The behaviour of some special sequences like \w, \W, \b,\s, \S will be made dependant on the current locale, i.e. the user's language, country aso.
re.M	re.MULTILINE	^ and $ will match at the beginning and at the end of each line and not just at the beginning and the end of the string
re.S	re.DOTALL	The dot "." will match every character **plus the newline**
re.U	re.UNICODE	Makes \w, \W, \b, \B, \d, \D, \s, \S dependent on Unicode character properties
re.X	re.VERBOSE	Allowing "verbose regular expressions", i.e. whitespace are ignored. This means that spaces, tabs, and carriage returns are not matched as such. If you want to match a space in a verbose regular expression, you'll need to escape it by escaping it with a backslash in front of it or include it in a character class. # are also ignored, except when in a character class or preceded by an non-escaped backslash. Everything following a "#" will be ignored until the end of the line, so this character can be used to start a comment.



To specify more than one of them, use `|` operator to connect them. For example, `re.search(pattern,string,flags=re.IGNORECASE|re.MULTILINE|re.UNICODE)`.

Thank you