Lex regular expressions

abc	: The abc string.
“abc”	: Same as above.
\”	: A double quote character.
\/	: A slash character.
[abc]	: a, b or c. Same as a|b|c.
[a-zA-Z]	: Any alphabetical character.
[1-9]	: A non-zero digit.
[+-]?	: +, - or neither. Same as +| − |ε .
(Wa)?(Luigi)	: Luigi or WaLuigi.
[0-9]+	: A non-empty sequence of digits.
[0-9]*	: Any number of digits in a row. Same as ([0-9]+)? .
[^a]	: Any character except a.
[^abc]	: Any character except a, b and c.
[ ]	: A space character.
[\t\n]	: Any form of whitespace. (A space, a tab or a line feed)
.	: Any character except \n. Same as [^\n].
\.	: The dot character.

[.] : A dot character.

‘\\n’ : ‘\n’

0x(0|([1-9a-fA-F][0-9a-fA-F]*))

A hexadecimal number starting with a 0x. Example: 0xFFFF or 0xabc22D.

Note that when writing regular expressions, you should avoid using whitespaces with the intent of making the code more readable.

The below statement is not a substitution to the one stated above.

0x (0 | ([1-9a-fA-F] [0-9a-fA-F] *))

(r1)/(r2) : r1, only if followed by r2.

This lookahead operator does not consume the lookahead characters. (r2 is tokenized separately.)

Example:

1. {integer}/[=]
2. “==”

Input: 100==20

100 is returned by rule 1.

== is returned by rule 2.

20 is not matched by rule 1, since it is not followed by an = character.