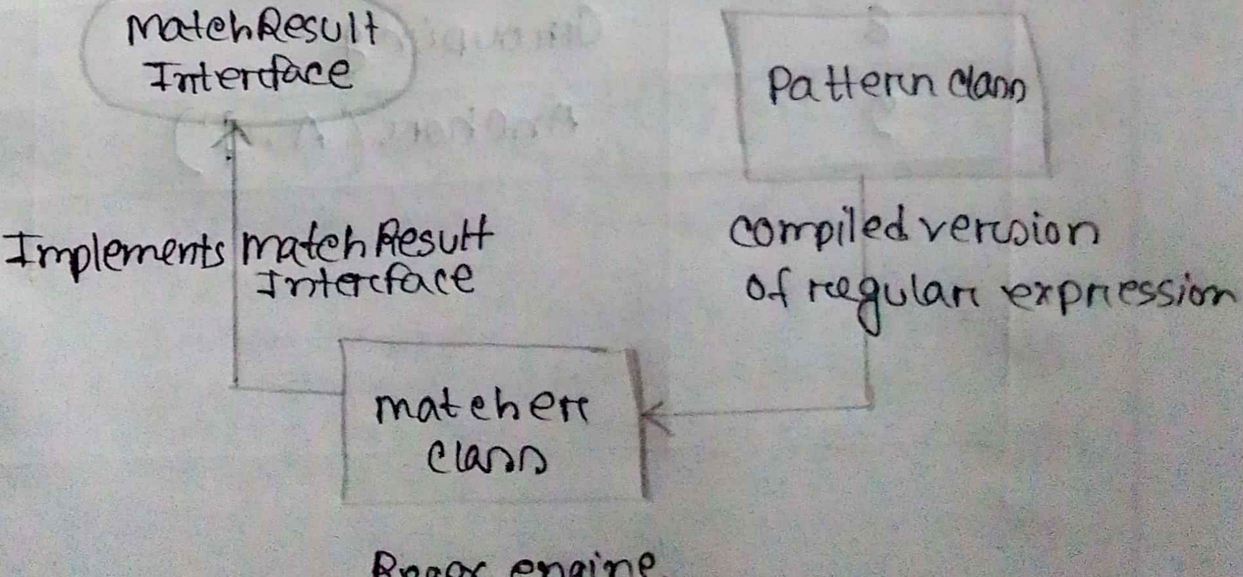
Regular Expressions

→ Regular Expression basically defines a search pattern pattern matching, or string matching. It is present in java.util.regex package. Java Regex API provides 1 interface and 3 classes. They are the following :

1. matchResult Interface
2. matcher class
3. pattern class
4. pattern syntex Exception class



MatchResult Interface
Pattern class
Implements matchResult Interface
compiled version of regular expression
matcher class
Regex engine

Regulare Expression

- Pattern p = Pattern . compiler (".e");
- matcher m = p . matcher ("Leading");
- boolean b = m . matches ();

Regex Basic

1	Common Use & How to start
2	Flag : g, i
3	Quantification(?, *, + {n})
4	Character (w, d, s, W, D, S)
5	Use of 3rd Bracket
6	Escape (\)
7	Logical or (|)
8	Grouping
9	Anchor (^, $)

Elements used to write Regular Expression

(*)	0 or more times (upto infinite)	ab*c will give ac, abc, abbc abbbc - - - and so on
(+)	Atleast 1 or more times (up to infinite)	ab+c will give abc, abbc, abbbc, - - - & so on.
{....}	for as many times as the value inside this bracket	{2} means that the preceding character is to be repeated 2 times. {min,} means the preceding character is matches min or more time.
wildcard (•)	The dot symbol can take the place of any other symbol	•* will tell the computer that any character can be used any number of times.
optional character (?)	0 or 1 time. may or may not be present.	"docx?" The '?' tells the computer that x may or may not be present in the name of file format
Caret '^'	The match must start at the beginning of the string or line	^\d{3} will match with patterns like "901" in "901-333-".
Doller '$'	The match must occur at the end of the string or before in at the end	-\d{3}$ will match with patterns like "-333" in "-901-333".

Character classess

\s	matches any whitespace char such as space and tab
\S	matches any non-whitespace characters
\d	matches any digit character.
\D	matches any non-digit character
\w	matches any word character
\W	matches any non-word character
[set of character]	Matches any single character in set-of-characters.

Regular Expression

→ [^set-of-character] Negation:

matches any sigle character that is not in set-of-characters. By default, the match is case sensitive.

Example: [^abc] will match any char except a, b, c

→ [first-last] character range: matches any single character in the range from first to last.

Example: [a-zA-Z] will match any char from a to z or A to Z.

→ The Escape Symbol (\): If you want to match for the actual '+' '.' etc charecters, add a backslash (\) before that character.

Example: \d+[\+-x*] \d+ will match patters like "2+2" & "3*9" in "(2+2)* 3*9".

→ Grouping characters ()

A set of different symbols of a regular expression can be grouped togethere to act as a single unit and behave as a block, for this, you need to wrap the regular expression in the parenthesis().

Example: ([A-Z]\w+) contains two different elements of the regex combined together. This expression will match any pattern containing uppercase letter.

→ Vartical Bar (|)

matches any one element separated by the vertical bar (|) character

Example: th(e|is|at) will match words - the, this and that.