

Realice una investigación en internet acerca de las diferentes herramientas que existen para simular o manejar expresiones regulares (al menos 2) incluya un lenguaje de programación, para cada herramienta investigue: características, forma de uso, elementos importantes, elabore cuadro con características:

Herramienta	Lenguaje de Programación	Características	Formas de uso	Elementos Importantes
Regex101	JavaScript	<p>Sistema de ayuda integrado con referencias de token.</p> <p>Explicación de expresiones regulares para ayudar a entenderlas.</p> <p>Exportador de código para PHP, Python y JavaScript.</p>	<p>Introducir la Regular Expression que queremos utilizar y automáticamente nos mostrará a la derecha el resultado sobre cómo sería la mejor forma de utilizarla en nuestro programa.</p>	<p>[], Los corchetes identifican a una clase de caracteres que siempre representa a un único carácter en un patrón de búsqueda.</p> <p>(), Los paréntesis identifican un grupo de caracteres formado por uno o varios caracteres y que pueden operarse unos dentro de los otros.</p>

Herramienta	Lenguaje de Programación	Características	Forma de uso	Elementos importantes
Txt2r online	JAVA, PHYTON	Herramienta online más antiguas para ayudarnos a crear expresiones regulares. Su objetivo es darnos ya creada una simple expresión, aunque necesita alguna optimización,	Cuando insertamos una cadena, despliega una tabla con distintas jerarquías, que deberemos clicar según el nivel de profundidad que queramos seleccionar en cada parte de la cadena. Tras esto generará un código en una variedad de lenguajes con el patrón que hayamos seleccionad o	<p>"\$"</p> <p>Representa el final de la cadena de caracteres o el final de la línea, si se utiliza el modo multi-línea. No representa un carácter en especial sino una posición.</p> <p>"*"</p> <p>El asterisco sirve para encontrar algo que se encuentra repetido 0 o más veces.</p>

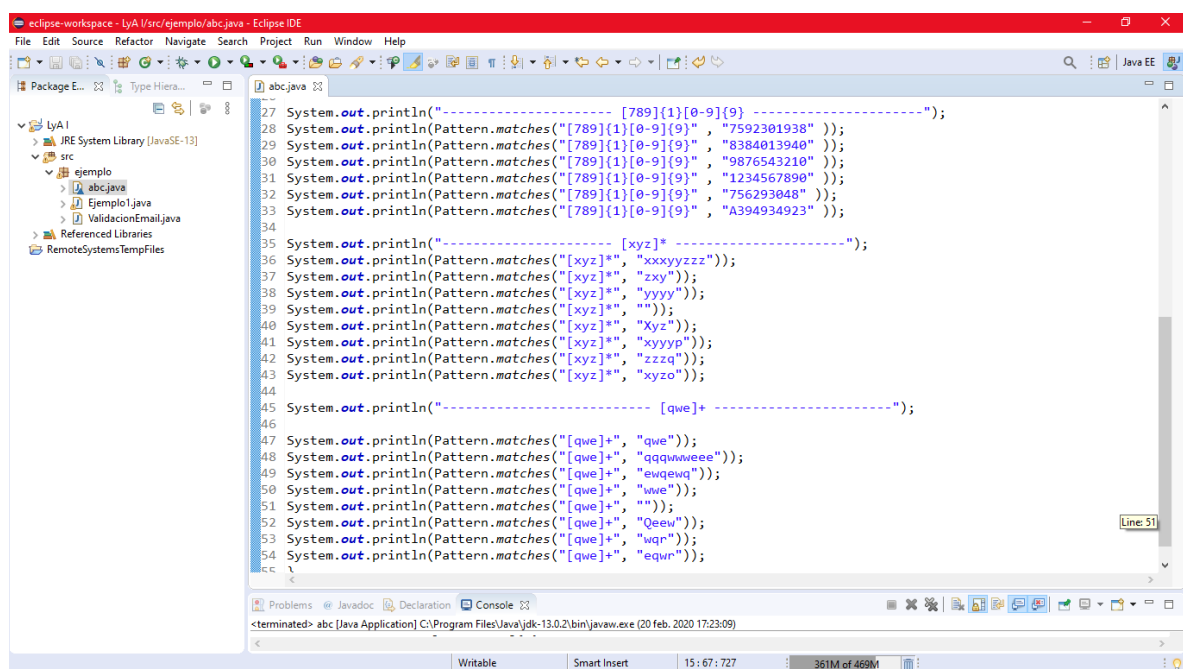
Considere un lenguaje de programación y obtenga la expresión regular para al menos 8 componentes léxicos, elabore tabla (componente léxico, descripción informal, expresión regular)

JAVA		
Componente léxico	Descripción informal	Expresión Regular
Palabra reservada Void, static ,new, null	void :indica que el método main no retorna ningún valor. New : Separa memoria para el nuevo objeto, Invoca el método de inicio de la clase llamado constructor, Retorna la referencia a un nuevo objeto. <ul style="list-style-type: none"> null : resulta ser un valor especial aplicado a un puntero (o referencia) usado para indicar que no se apunta a un objeto o dato válidos. 	<pre>public class Hello1 { public static void main (String args[]) { System.out.println("Hello World!"); }</pre>
Identificadores (x, y , z, a, z,_, \$:solo son para basarme).	Tienen que empezar con una letra.	Tu_re_da Usd\$
Operadores Aritméticos +,-,*,/,%	Mas, Menos, Asterisco, División, por ciento	+,-,*,/,%
Operadores lógicos !=, ll,&&, not ,==	Es distinto, Operador or(), Operador and(y), Es igual	j=,ll,&&
Operadores Relacionales >,>=,<,<=,==, j=, char , boolean	Mayor, Mayor o igual, Menor, Menor o igual, Es igual	>,>=,<,<=
Separadores (), {} , ; , . , []	Paréntesis,corchetes Punto y coma,punto,llaves	(),{}
Números 0, 1, 2, 3, ..., 9	Números	(0-9)*

Comentarios // /* /**	// Comentario de una sola línea. /* Comentario de una o más líneas. /** comentario de documentación, una o más líneas	// todos los leds prenden // /* patricia castellanos*/
--------------------------------	--	--

Utilizando el lenguaje de programación que investigo para el manejo de expresiones regulares, pruebe al menos 5 expresiones regulares de componentes léxicos del punto anterior y genere un programa que valide y utilice las expresiones generadas para cadenas validas e invalidas (pruebe al menos 3 ejemplos de cada expresión regular), considere manejo errores y diseño lógico del programa acorde al paradigma utilizado en el lenguaje de programación, anexe pantallas de ejecución de cada componente probado y elabore tabla con las cadenas validas e invalidas:

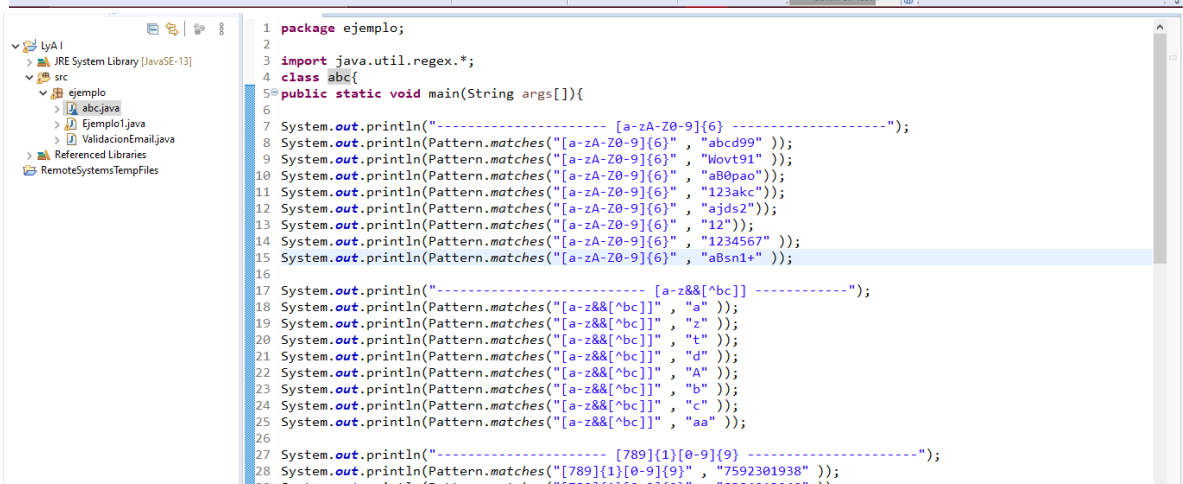
- Este es el código en el que introduces una expresión regular y seguidamente la cadena con el fin de determinar si es válida o inválida.



```

27 System.out.println("----- [789]{1}[0-9]{9} -----");
28 System.out.println(Pattern.matches("[789]{1}[0-9]{9}", "7592301938" ));
29 System.out.println(Pattern.matches("[789]{1}[0-9]{9}", "8384013940" ));
30 System.out.println(Pattern.matches("[789]{1}[0-9]{9}", "9876543210" ));
31 System.out.println(Pattern.matches("[789]{1}[0-9]{9}", "1234567890" ));
32 System.out.println(Pattern.matches("[789]{1}[0-9]{9}", "756293048" ));
33 System.out.println(Pattern.matches("[789]{1}[0-9]{9}", "A394934923" ));
34
35 System.out.println("----- [xyz]* -----");
36 System.out.println(Pattern.matches("[xyz]*", "xxxxyzzzz"));
37 System.out.println(Pattern.matches("[xyz]*", "zxy"));
38 System.out.println(Pattern.matches("[xyz]*", "yyyy"));
39 System.out.println(Pattern.matches("[xyz]*", ""));
40 System.out.println(Pattern.matches("[xyz]*", "Xyz"));
41 System.out.println(Pattern.matches("[xyz]*", "xyyyp"));
42 System.out.println(Pattern.matches("[xyz]*", "zzzz"));
43 System.out.println(Pattern.matches("[xyz]*", "xyzzy"));
44
45 System.out.println("----- [qwe]+ -----");
46
47 System.out.println(Pattern.matches("[qwe]+", "qwe"));
48 System.out.println(Pattern.matches("[qwe]+", "qqqnnwweee"));
49 System.out.println(Pattern.matches("[qwe]+", "ewqewq"));
50 System.out.println(Pattern.matches("[qwe]+", "wwe"));
51 System.out.println(Pattern.matches("[qwe]+", ""));
52 System.out.println(Pattern.matches("[qwe]+", "Qeew"));
53 System.out.println(Pattern.matches("[qwe]+", "wqn"));
54 System.out.println(Pattern.matches("[qwe]+", "eqwr"));
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99

```



```

1 package ejemplo;
2
3 import java.util.regex.*;
4 class abc{
5     public static void main(String args[]){
6
7         System.out.println("----- [a-zA-Z0-9]{6} -----");
8         System.out.println(Pattern.matches("[a-zA-Z0-9]{6}", "abcd99" ));
9         System.out.println(Pattern.matches("[a-zA-Z0-9]{6}", "Wovt91" ));
10        System.out.println(Pattern.matches("[a-zA-Z0-9]{6}", "aB0pao"));
11        System.out.println(Pattern.matches("[a-zA-Z0-9]{6}", "123akc"));
12        System.out.println(Pattern.matches("[a-zA-Z0-9]{6}", "ajds2"));
13        System.out.println(Pattern.matches("[a-zA-Z0-9]{6}", "12"));
14        System.out.println(Pattern.matches("[a-zA-Z0-9]{6}", "1234567" ));
15        System.out.println(Pattern.matches("[a-zA-Z0-9]{6}", "aBsn1+"));
16
17        System.out.println("----- [a-zA-Z0-9]{6} -----");
18        System.out.println(Pattern.matches("[a-zA-Z0-9]{6}", "a" ));
19        System.out.println(Pattern.matches("[a-zA-Z0-9]{6}", "z" ));
20        System.out.println(Pattern.matches("[a-zA-Z0-9]{6}", "t" ));
21        System.out.println(Pattern.matches("[a-zA-Z0-9]{6}", "d" ));
22        System.out.println(Pattern.matches("[a-zA-Z0-9]{6}", "A" ));
23        System.out.println(Pattern.matches("[a-zA-Z0-9]{6}", "b" ));
24        System.out.println(Pattern.matches("[a-zA-Z0-9]{6}", "c" ));
25        System.out.println(Pattern.matches("[a-zA-Z0-9]{6}", "aa" ));
26
27        System.out.println("----- [789]{1}[0-9]{9} -----");
28        System.out.println(Pattern.matches("[789]{1}[0-9]{9}", "7592301938" ));
29        System.out.println(Pattern.matches("[789]{1}[0-9]{9}", "8384013940" ));
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99

```

- Cuando se ejecuta el programa, envía un mensaje en la consola ya sea “true” o “false” en donde true indica que la cadena introducida junto a la expresión regular definida previamente es completamente válida y false indica que esta cadena no es válida y se puede deber a que no cumple con las reglas de la expresión regular.

```

eclipse-workspace - lyA /src/ejemplo/abc.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
<terminated> abc [Java Application] C:\Program Files\Java\jdk-13.0.2\bin\javaw.exe (20 feb. 2020 21:32:54)
----- [a-zA-Z0-9]{6} -----
true
true
true
true
false
false
false
false
----- [a-zA-Z^bc] -----
true
true
true
true
false
false
false
false
----- [789]{1}[0-9]{9} -----
true
true
true
false
false
false
false
----- [xyz]* -----
true
true
true
true
false
false
false

```

```

----- [789]{1}[0-9]{9} -----
true
true
true
true
false
false
false
----- [xyz]* -----
true
true
true
true
false
false
false
----- [que]+ -----
true
true
true
true
false
false
false

```

- A continuación, en base al programa realizado, se muestra una tabla con la expresión regular utilizada, una descripción informal y cadenas que pueden ser válidas así como las inválidas:

EXPRESIÓN REGULAR	DESCRIPCIÓN	CADENAS VALIDAS	CADENA NO VALIDAS
[a-zA-Z0-9]{6}	Cualquier letra minúscula entre la a y la z y/o letra mayúscula entre la A y Z y/o número entre el 0 y 9, todos en una cadena solamente de 6 caracteres.	abcd99, Wovt91, aB0pao, 123akc	ajds2, 12, 1234567, aBsn1+
[a-z&&[^bc]]	Una letra entre la a y z exceptuando la b y c.	a, z, t, d	A, b, c, aa
[789]{1}[0-9]{9}	El número 7, 8 o 9 seguido de una cadena de 9 números que estén entre el 0 y 9	7592301938, 8384013940, 9876543210	1234567890, 756293048, A394934923
[xyz]*	Cero o más x's, y's y/o z's	xxxyyzzz, xyz, yyy, ε	Xyz, zzzq, xyyp,xyzo
[qwe]+	Una o más q's, w's y/o e's	qwe, qqwwweee, ewqewq, wwe	ε, Qeew, wqr, eqwr

DESCRIPCIÓN DE LA PRÁCTICA:

Descripción de Mayra Castillo Montenegro:


Como Trabajo en equipo me toco realizar la investigación de las herramientas que utilizan o manejan las expresiones regulares, el lenguaje de programación que utilizan sus diferentes características, forma de uso y sus elementos más importantes realizando una tabla con cada una de estas opciones.

Descripción de Patricia Fernanda Castellanos Ruiz:

La parte que me toco a mi constaba de hacer una tabla describiendo 8 componentes léxicos de algún lenguaje de los que habíamos hecho, en este caso use el lenguaje de java , en la cual hice una tabla con su descripción informal, y expresión regular , batalle mucho al saber cómo hacer las expresiones regulares de java , pero le pedí ayuda a la maestra y pude comprender mejor como hacerlas, al igual me guie viendo información en páginas pero no anote nada de ellas , solo me base en saber información y así pude terminar mi parte.

Descripción de Jesús Eduardo Silva Vázquez:

Para el tercer punto de la práctica realicé un programa en Java utilizando Eclipse capaz de analizar si una cadena es válida o inválida en base a la expresión regular impuesta. Para ello se necesitaba una librería llamada “regex.jar” (Fig.1), esta librería externa fue importada dentro del paquete del código para que nuestro código en Java pudiera comprender las expresiones regulares y por lo tanto analizar cadenas.

 [regex/regex.jar.zip\(201 k\)](#)

The download jar file contains the following class files or Java source files.

```
java.util.regex.PreviousMatch.class
java.util.regex.Quantifier.class
java.util.regex.QuantifierSet.class
java.util.regex.RangeSet.class
java.util.regex.RelAltGroupQuantifierSet.class
java.util.regex.RelCompositeGroupQuantifierSet.class
java.util.regex.ReluctantAltQuantifierSet.class
java.util.regex.ReluctantCompositeQuantifierSet.class
java.util.regex.ReluctantGroupQuantifierSet.class
java.util.regex.ReluctantQuantifierSet.class
```

Fig.1 Librería regex para el manejo de expresiones regulares en Java.

CONCLUSIONES

Conclusión personal de Mayra Castillo Montenegro

En esta práctica se aprendió mas acerca de las expresiones regulares e incluso se conocieron gracias a la investigación herramientas que trabajan con expresiones regulares que en lo personal me parece interesante ya que son fácil de usar y la mayoría son online.

Conclusión personal de Patricia Fernanda Castellanos Ruiz:

En esta segunda practica en expresiones regulares en lenguajes de programación, investigamos sobre 2 lenguajes de programación que fueron javaScript, java phyton , en donde también pusimos 2 herramientas donde también pusimos sus características, forma de uso y elementos importantes , y después hicimos una tabla con los componentes léxicos de java , con su descripción informal y su expresión regular y en el inciso siguiente se hizo en el lenguaje de programación un se realizó un programa en Java utilizando la librería Regex para el manejo de expresiones regulares y determinar si una cadena previamente introducida es válida (true) o inválida (false) esto con el fin de comprender como funciona un compilador léxico, más que nada vimos las expresiones regulares al cómo usarlas e identificarlas.

Conclusión personal de Jesús Eduardo Silva Vázquez:

Las expresiones regulares son bastante útiles para el manejo de textos y la búsqueda o definición de patrones de caracteres; la mejor forma de introducirse al tema de expresiones regulares es utilizando algún lenguaje de programación del que ya se tenga conocimiento, debido a que al contar con un previo conocimiento respecto a la programación básica y ser capaz crear programas, se puede combinar con el tema de expresiones regulares y facilitar su manejo.

BIBLIOGRAFIAS:

<https://www.javatpoint.com/java-regex>

[https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Regular Expressions](https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Regular_Expressions)

<https://regex101.com/>

<https://regexr.com/>

<http://www.txt2re.com/index-vb6.php3?s=arnab&5&7&4>

https://www.tutorialspoint.com/java/java_regular_expressions.htm