EXPRESIONES REGULARES

Iván Espinosa Fourquet

Raúl García Pablos

1. Crea una expresión regular que sirva para validar un DNI
2. Crea una expresión regular que sirva para validar una fecha (dd/mm/aaa)
3. Crea una expresión regular que sirva para validar un número real
4. Crea una expresión regular que sirva para buscar en el texto todas las fechas (“díaXX” de “mes” de “año”)
5. Crea una expresión regular que sirva para buscar en el texto todas las palabras acentuadas

SOLUCIONES (ejercicios 1. 2. y 3. 🡪 ValidacionesSimples.java ; ejercicio 4. 🡪 FechasBOE.java ; ejercicio 5. 🡪 Acentuadas.java)

Ruth Hernández Quitián

**Validación de nombres de usuario**

1. Dado el siguiente listado de nombres de usuarios, habrá que contabilizar cuántos de ellos son válidos. Cada elemento del listado estará separado del siguiente por un punto y coma y un salto de línea.

Los nombres de usuario válidos tendrán las siguientes características:

* Los nombres de usuario estarán compuestos por caracteres alfanuméricos.
* También estará permitido el uso de la barra baja (\_) y el guion (-).
* El nombre de usuario tendrá que empezar y terminar por un carácter alfanumérico.
* El nombre de usuario tendrá que tener entre 5 y 20 caracteres.

Texto: [usuarios regex.txt](usuarios%20regex.txt)

Expresión regular:h"(^|\\n)[a-zA-Z0-9]([\_-]|[a-zA-Z0-9]){3,18}[a-zA-Z0-9](;|$)"

(^|\\n) 🡪 Nos aseguramos de coger el primer elemento del listado, el resto de elementos darán comienzo en el salto de línea.

[a-zA-Z0-9] 🡪 El primer carácter ha de ser un carácter alfanumérico.

([\_-]|[a-zA-Z0-9]){3,18}🡪 Los siguientes caracteres han de ser caracteres alfanuméricos o bien un guion o una barra baja. Habrá un número variable de caracteres entre 3 y 18, para ajustarlo a lo indicado teniendo en cuenta que el primer y el último se determinan de forma independiente.

[a-zA-Z0-9] 🡪 El último carácter ha de ser un carácter alfanumérico.

(;|$)"🡪 Nos aseguramos de coger el último elemento del listado y de dar por terminado el resto de elementos al llegar al punto y coma separador.

Resultado: Hay 49 nombres de usuario válidos en el listado.

**Validación de contraseñas**

1. Dado el siguiente listado de contraseñas, contabilizar cuántas de ellas son seguras. Cada elemento del listado estará separado del siguiente por un punto y coma y un salto de línea.

Las contraseñas seguras tendrán las siguientes características:

* Al menos un número.
* Al menos una letra minúscula.
* Al menos una letra mayúscula.
* Al menos uno de los caracteres especiales [\*.!@#$%^&(){}[]:;<>,.?/~\_+-=|\]
* Al menos 8 caracteres, pero menos de 32.

Texto: [contraseñas regex.txt](contraseñas%20regex.txt)

**Lexema**

1. En el siguiente texto, contabilizar cuántas veces aparece una palabra con la misma raíz que la palabra “oler” (¡tener en cuenta que es un verbo irregular!).
2. Realizar una lista con las palabras que hayan aparecido (no incluir las repetidas).

Texto: [lexema regex.txt](lexema%20regex.txt)

**Facturación**

En el siguiente texto, extraer el precio total de cada factura. Contabilizar los ingresos totales en cada una de las divisas (dólar $, euro €, libra esterlina £). ¿Cuál de los tres mercados es el más rentable?

Texto: [Regex Factura.xlsx](Regex%20Factura.xlsx)

Referencias

<https://www.ocpsoft.org/tutorials/regular-expressions/password-regular-expression/>

<https://mkyong.com/regular-expressions/how-to-validate-username-with-regular-expression/#:~:text=Username%20consists%20of%20alphanumeric%20characters,the%20first%20or%20last%20character.&text=The%20number%20of%20characters%20must%20be%20between%205%20to%2020>.

<https://dle.rae.es/oler?m=form>

<https://dle.rae.es/olor?m=form>