**EXPRESIONES REGULARES EN PROGRAMACIÓN**

Iván Espinosa Fourquet

Raúl García Pablos

Ruth Hernández Quitián

1ºC DAM

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)

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**:

**(^|\\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.

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)

**Expresión regular**:h

**(^|\\n)(?=.\*[0-9])(?=.\*[a-z])(?=.\*[A-Z])(?=.\*[\*.!@$%^&(){}\\[\\]:;<>,.?/~\_+-=|\\\\]).{8,32}(;|$)**

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

**(?=.\*[0-9])** 🡪 Al menos un carácter ha de ser un dígito.

**(?=.\*[a-z])** 🡪 Al menos un carácter ha de ser una letra minúscula.

**(?=.\*[A-Z])** 🡪 Al menos un carácter ha de ser una letra mayúscula.

**(?=.\*[\*.!@$%^&(){}\\[\\]:;<>,.?/~\_+-=|\\\\]).** 🡪 Al menos un carácter ha de ser un carácter especial.

.**{8,32}** 🡪 Cualquier carácter se repetirá entre 8 y 32 veces

**(;|$)**🡪 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 5 contraseñas seguras en el listado.

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!).

**Expresión regular**:

**"(^|\\W)(ol|huel|Ol|Huel)([a-z]+)(**[**\\W|$**](file:///\\W|$)**)"**

**(^|\\W)** 🡪 Nos aseguramos de coger la primera palabra de la línea, el resto de palabras estarán siempre precedidas por algún signo de puntuación, espaciados…

**(ol|huel|Ol|Huel)**🡪 Insertamos la raíz de la palabra, al ser irregular tendrá dos posibles formas, tenemos en cuenta que puede aparecer en mayúscula y también sería válida.

**([a-z]+)** 🡪 La raíz irá seguida de una o más letras minúsculas.

**(**[**\\W|$**](file:///\\W|$)**)**🡪 Nos aseguramos de coger la última palabra de la línea, el resto de palabras finalizarán al aparecer algún signo de puntuación, espaciados…

**Resultado:**

Hay 21 palabras que comparten raíz con la palabra “oler” en el texto dado.

1. Contabilizar el número de palabras sin repetir que aparecen en la búsqueda del ejercicio anterior y elaborar una lista con las mismas.

**Resultado:**

Hay 7 palabras en total que comparten raíz con la palabra “oler” si eliminamos las repetidas. Son las siguientes:

1. oler

2. olores

3. olor

4. huelo

5. huele

6. olfato

7. oliendo

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

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

**Expresión regular**:

Dólar:

**"\\$[0-9]\*,[0-9]\*\\r"**

**\\$**🡪 El símbolo dólar aparece antes de la cantidad, nos aseguramos de escapar el símbolo.

**[0-9]\*,**🡪 Habrá una cantidad de dígitos del 0 al 9 variable, y después la coma decimal.

**[0-9]\*** 🡪 Habrá una cantidad de dígitos del 0 al 9 variable tras la coma decimal.

[**\\r**](file:///\\W|$)🡪 Todas las cantidades estarán precediendo a un retorno de carro.

Euro:

**"€\\s[0-9]\*,[0-9]\*\\r"**

**€\\s**🡪 El símbolo euro aparece antes de la cantidad separado por un espacio.

**[0-9]\*,**🡪 Habrá una cantidad de dígitos del 0 al 9 variable, y después la coma decimal.

**[0-9]\*** 🡪 Habrá una cantidad de dígitos del 0 al 9 variable tras la coma decimal.

[**\\r**](file:///\\W|$)🡪 Todas las cantidades estarán precediendo a un retorno de carro.

Libra esterlina:

**"£[0-9]\*,[0-9]\*\\r"**

**£**🡪 El símbolo de la libra aparece antes de la cantidad.

**[0-9]\*,**🡪 Habrá una cantidad de dígitos del 0 al 9 variable, y después la coma decimal.

**[0-9]\*** 🡪 Habrá una cantidad de dígitos del 0 al 9 variable tras la coma decimal.

[**\\r**](file:///\\W|$)🡪 Todas las cantidades estarán precediendo a un retorno de carro.

**Resultado apartado A:**

El euro es el mercado más rentable.

Total ingresado en euros: 98126.04999999997 €.

Total ingresado en dólares: 91325.85999999997 $.

Total ingresado en libras: 16422.260000000002 £.

**Resultado apartado B:**

Hay 282 operaciones realizadas en dólares.

Hay 288 operaciones realizadas en euros.

Hay 47 operaciones realizadas en libras.

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>