Ejercicio 11 – Expresiones Regulares

2. Analizar las siguientes expresiones regulares.

a) var miExpReg = /as?.a/

Que contenga una 'a' seguida de una 's' cero o una vez luego cualquier catacter y una a

b) var re = /ab+c/

Que contenga una ab una o varias veces y luego una c

c) var re = new RegExp("ab+c")

que tenga tanto ab como una c

d) /([.*+?^${}()|\[\]\/\\])/g

Puede tener cualquier caracter una o mas veces, termina con corchete, dos barras otor corchete y además no se detiene al encontrar la primera coincidencia

e) /\w+\s/g

Que comience por cualquier digito o letra minuscula o mayuscula, despues un solo espacio en blanco y lo comprueba en toda la cadena

f) ([B-D]|[F-H]|[J-N]|[P-T]|[V-Z])

Una letra mayuscula de la b a la d, otra mayuscula de la f a la h, otra mayuscula de la j a la n, otra mayuscula de la p a la t y otra mayuscula de la v a la z

g) /\S+@\S+\.\S+/

Un solo caracter que no sea espacio en blanco, seguido de arroba luego otor caracter no siendo este espacio en blanco, un punto y un caracter que no sea espacio en blanco

h) (^[0-9\s\+\-])+$/

Comienza con un numero de 0 a 9, luego un espacio en blanco, un guion y termine por barra

i) /^(.+\@.+\..+)$/

Comienza con uno o mas caracteres luego una arroba una o mas veces luego un punto seguido de cualquier caracter una o mas veces termina todo esto al final

j) / [a-z0-9!#$%&'*+/=?^_`{|}~-]+(?:\.[a-z0-9!#$%&'*+/=?^_`{|}~-]+)*@(?:[a-z0-9](?:[a-z0-9-]*[a-z0-9])?\.)+[a-z0-9](?:[a-z0-9-]*[a-z0-9])?/

Empieza por una letra o un numero excluyendo simbolos luego sige de cero a una veces dos puntos luego de nuevo una letra o un numero de cero a mas veces sin ningun simbolo despues una arroba y cero o una vez los dos puntos luego una o mas letras o numero del cero al nueve, todas las letras en minuscula

3. Creación de una expresión regular (TEORIA)

Par crear una expresión regular, puede utilizarse dos métodos:

a) La primera opción compila la expresión regular cuando se evalúa el script, por lo que es mejor cuando la expresión regular es una constante (delimitada por barras) y no va a variar a lo largo de la ejecución del programa.

```
exp_reg1 = /^[0-9]+/;
```

La variable se convierte en una variable del tipo expresión regular, por tanto, puede usarse con ella el método test para validar la cadena.

```
    if(exp_reg1.test(“123”)==false)
```

b) La segunda opción compila la expresión regular en tiempo de ejecución (guardada en una variable de tipo cadena o en un campo de un formulario). Aquí los delimitadores son las comillas dobles, no las barras.

```
exp_reg2 = new RegExp("^[0-9]+");
// Ahora exp_reg2 es una variable que contiene una expresión regular.
exp_reg3 = new RegExp(formu.campo1.value);
// exp_reg3 tendrá como expresión regular el contenido del campo campo1 del formulario formu.
exp_reg4 = new RegExp(cadena1);
```

```
// exp_reg4 tendrá como expresión regular el contenido de la variable de cadena cadena1.
if(exp_reg3.test(“123”)==false)
// Ahora podrá usarse el método test en las variables.
```