

# Práctica 03: Expresiones regulares

Computabilidad y Algoritmia  
Grado en Ingeniería Informática  
Universidad de La Laguna

Alejandro Rodríguez Rojas  
[alu0101317038@ull.edu.es](mailto:alu0101317038@ull.edu.es)

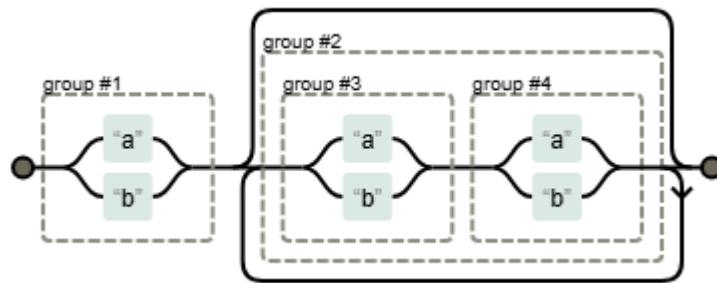
# Índice

<b>1. Ejercicios sobre operadores básicos</b>	<b>3</b>
1.1. Cadenas sobre el alfabeto {a, b} con longitud impar	3
1.2. Cadenas sobre el alfabeto {a, b} con longitud igual a 5.	4
1.3. Cadenas sobre el alfabeto {a, b, c} con una “a” en la antepenúltima posición.	
5	
1.4. Cadenas sobre el alfabeto {a, b} con número de “a’s” par o número de “b’s” impar.	6
1.5. Cadenas w sobre el alfabeto {0, 1} tales que $2 \leq  w  \leq 5$	7
1.6. Cadenas sobre el alfabeto {0, 1} con longitud múltiplo de 3	8
1.7. Cadenas sobre el alfabeto {0, 1} con una longitud que no sea múltiplo de 3.	9
1.8. Cadenas w sobre el alfabeto {0, 1} tal que $w = 0^n1^m$ con $n + m$ impar.	10
1.9. Cadenas sobre el alfabeto {0, 1, 2, 3, 4, 5, 6, 7, 8, 9} que tengan como máximo dos ceros.	11
1.10. Cadenas sobre el alfabeto {x, y, z} que no contengan dos símbolos x consecutivos.	13
<b>2. Ejercicios sobre operadores extendidos</b>	<b>14</b>
2.1. Direcciones de correos electrónicos de estudiantes de la Universidad de La Laguna.	14
2.2. Palabras que terminen por una vocal.	15
2.3. Números enteros.	15
2.4. Texto que se encuentre entre paréntesis.	16
2.5. Código postal en España.	17
2.6. Palabras que contienen sólo letras mayúsculas.	17
2.7. Números de teléfono en formato prefijo XXX-XXX-XXX, donde el prefijo del país puede indicarse empezando por 00 o bien con un símbolo +; por ejemplo, 0034 o +34 para España.	18
2.8. Fecha en formato DD/MM/AAAA	18
2.9. Palabras de al menos 10 letras de longitud.	19
2.10. Palabras que terminen con “ing” o “ed”.	19
<b>3. Modificación</b>	<b>20</b>
3.1. Sentencias class en C++	20
3.2. Lenguaje binario con número par de 1's	20

# 1. Ejercicios sobre operadores básicos

## 1.1. Cadenas sobre el alfabeto {a, b} con longitud impar

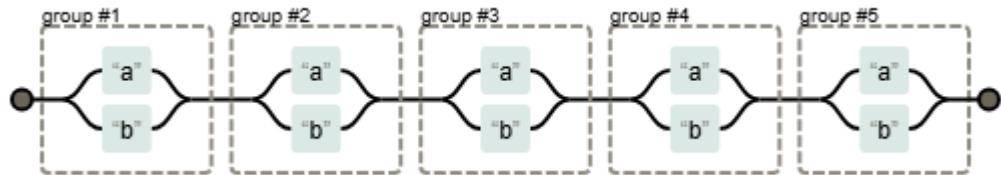
- Expresión regular:  $(a|b)((a|b)(a|b))^*$



- Cadenas que pertenecen al lenguaje:
  - $w_1 = a$
  - $w_2 = aba$
  - $w_3 = babaa$
  - $w_4 = aaaaaaaaa$
  - $w_5 = bbabbba$
- Cadenas que no pertenecen al lenguaje:
  - $w_6 = ba$
  - $w_7 = abab$
  - $w_8 = bbaabb$
  - $w_9 = aa$
  - $w_{10} = bababa$

## 1.2. Cadenas sobre el alfabeto {a, b} con longitud igual a 5.

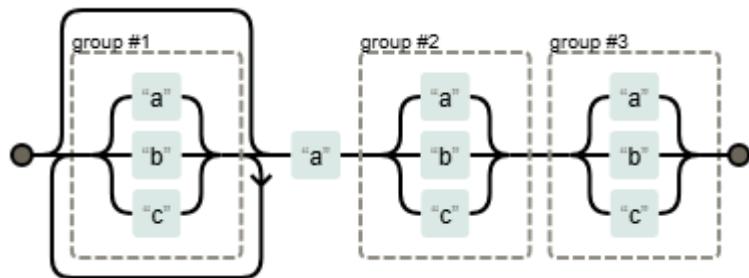
- Expresión regular:  $(a|b)(a|b)(a|b)(a|b)(a|b)$



- Cadenas que pertenecen al lenguaje:
  - $w_1 = \text{aaaaa}$
  - $w_2 = \text{ababa}$
  - $w_3 = \text{bbabb}$
  - $w_4 = \text{bbbbbb}$
  - $w_5 = \text{aabba}$
- Cadenas que no pertenecen al lenguaje:
  - $w_6 = \text{a}$
  - $w_7 = \text{bb}$
  - $w_8 = \text{bbb}$
  - $w_9 = \text{ababababab}$
  - $w_{10} = \text{aabb}$

### 1.3. Cadenas sobre el alfabeto {a, b, c} con una “a” en la antepenúltima posición.

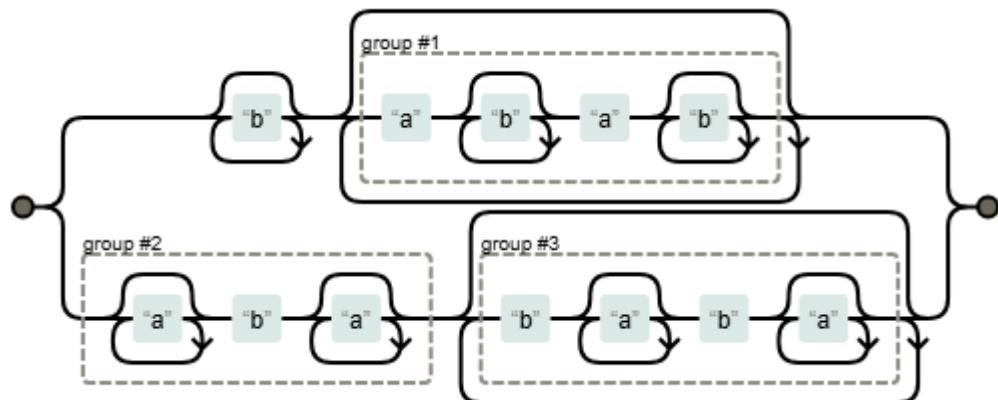
- Expresión regular:  $(a|b|c)^* a(a|b|c)(a|b|c)$



- Cadenas que pertenecen al lenguaje:
  - $w_1 = abc$
  - $w_2 = abbabb$
  - $w_3 = cabb$
  - $w_4 = aaa$
  - $w_5 = bacc$
- Cadenas que no pertenecen al lenguaje:
  - $w_6 = ccc$
  - $w_7 = bca$
  - $w_8 = a$
  - $w_9 = bbbca$
  - $w_{10} = aacccc$

#### 1.4. Cadenas sobre el alfabeto {a, b} con número de “a’s” par o número de “b’s” impar.

- Expresión regular:  $b^*(ab^*ab^*)^* \mid (a^*ba^*)(ba^*ba^*)^*$

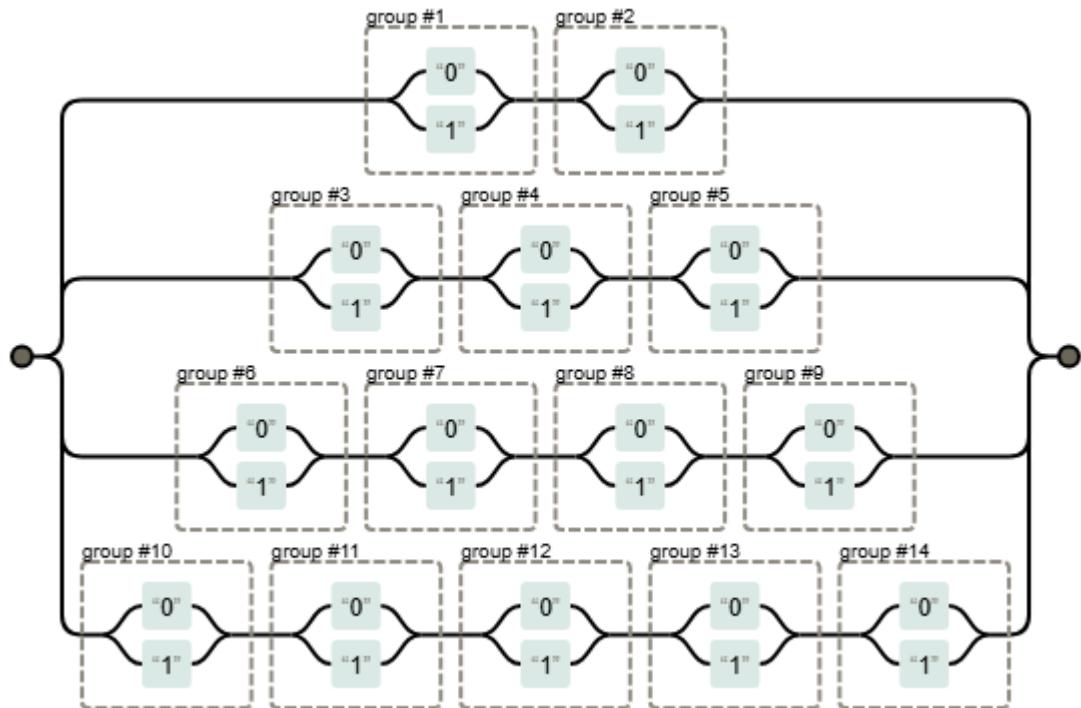


- Cadenas que pertenecen al lenguaje:
  - $w_1 = baa$
  - $w_2 = aabb$
  - $w_3 = bbba$
  - $w_4 = aa$
  - $w_5 = b$
- Cadenas que no pertenecen al lenguaje:
  - $w_6 = a$
  - $w_7 = bb$
  - $w_8 = abb$
  - $w_9 = aaaaabbbbbbb$
  - $w_{10} = bbaaabbb$

## 1.5. Cadenas $w$ sobre el alfabeto $\{0, 1\}$ tales que $2 \leq |w| \leq 5$

- Expresión regular:

$(0|1)(0|1) \mid (0|1)(0|1)(0|1) \mid (0|1)(0|1)(0|1)(0|1) \mid (0|1)(0|1)(0|1)(0|1)(0|1)$



- Cadenas que pertenecen al lenguaje:

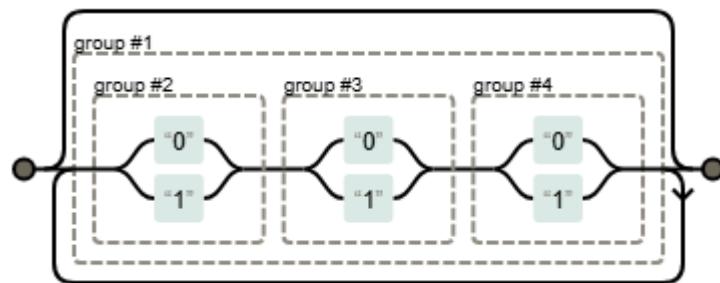
- $w_1 = 01$
- $w_2 = 111$
- $w_3 = 0011$
- $w_4 = 001$
- $w_5 = 10011$

- Cadenas que no pertenecen al lenguaje:

- $w_6 = 1$
- $w_7 = 0$
- $w_8 = 1111110000$
- $w_9 = 0000000000$
- $w_{10} = 101010101010110101010$

## 1.6. Cadenas sobre el alfabeto {0, 1} con longitud múltiplo de 3

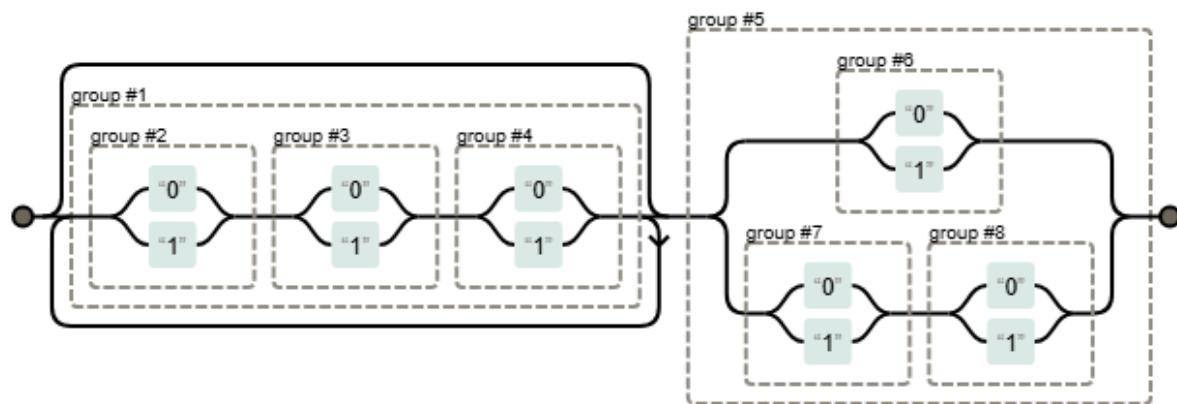
- Expresión regular:  $((0|1)(0|1)(0|1))^*$



- Cadenas que pertenecen al lenguaje:
  - $w_1 = 000$
  - $w_2 = 111000$
  - $w_3 = 101010101$
  - $w_4 = 001100111000$
  - $w_5 = 110$
- Cadenas que no pertenecen al lenguaje:
  - $w_6 = 1$
  - $w_7 = 11$
  - $w_8 = 1111$
  - $w_9 = 0110011$
  - $w_{10} = 0111$

1.7. Cadenas sobre el alfabeto {0, 1} con una longitud que no sea múltiplo de 3.

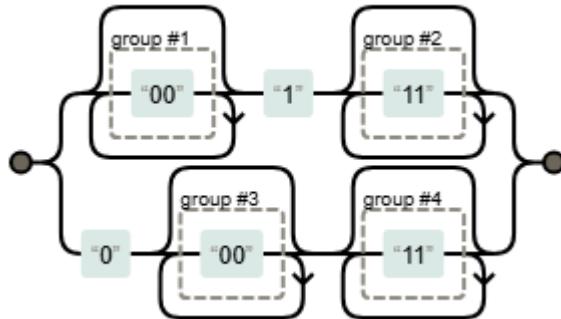
- Expresión regular:  $((0|1)(0|1)(0|1))^* ((0|1)|(0|1)(0|1))$



- Cadenas que pertenecen al lenguaje:
    - $w1 = 0$
    - $w2 = 1$
    - $w3 = 11$
    - $w4 = 0011$
    - $w5 = 0000110$
  - Cadenas que no pertenecen al lenguaje:
    - $w6 = 000$
    - $w7 = 101010$
    - $w8 = 11111111$
    - $w9 = 111$
    - $w10 = 101$

1.8. Cadenas  $w$  sobre el alfabeto  $\{0, 1\}$  tal que  $w = 0^n1^m$  con  $n + m$  impar.

- Expresión regular:  $(00)^*1(11)^* \mid 0(00)^*(11)^*$

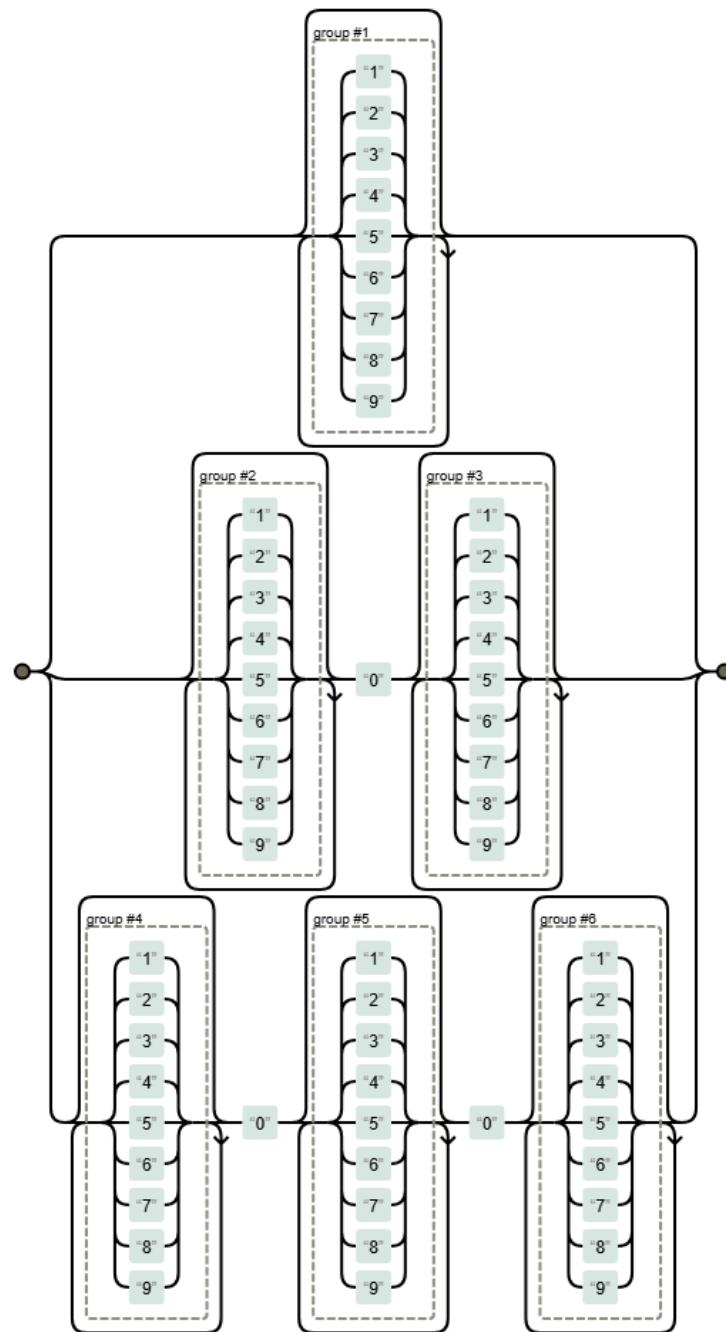


- Cadenas que pertenecen al lenguaje:
  - $w_1 = 0$
  - $w_2 = 1$
  - $w_3 = 0001111$
  - $w_4 = 00111$
  - $w_5 = 011$
- Cadenas que no pertenecen al lenguaje:
  - $w_6 = 0011$
  - $w_7 = 0111$
  - $w_8 = 0101$
  - $w_9 = 1111$
  - $w_{10} = 001111$

1.9. Cadenas sobre el alfabeto  $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$  que tengan como máximo dos ceros.

- Expresión regular:

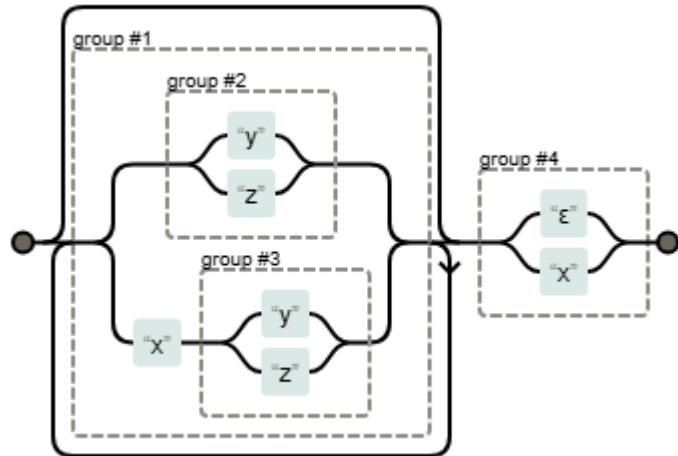
$$(1|2|3|4|5|6|7|8|9)^* \mid (1|2|3|4|5|6|7|8|9)^* 0 (1|2|3|4|5|6|7|8|9)^* \mid \\ (1|2|3|4|5|6|7|8|9)^* 0 (1|2|3|4|5|6|7|8|9)^* 0 (1|2|3|4|5|6|7|8|9)^*$$



- Cadenas que pertenecen al lenguaje:
  - $w_1 = 12345$
  - $w_2 = 54216023$
  - $w_3 = 00$
  - $w_4 = 10103234$
  - $w_5 = 4443$
- Cadenas que no pertenecen al lenguaje:
  - $w_6 = 000$
  - $w_7 = 800709$
  - $w_8 = 02220220$
  - $w_9 = 01001$
  - $w_{10} = 003333300$

1.10. Cadenas sobre el alfabeto  $\{x, y, z\}$  que no contengan dos símbolos x consecutivos.

- Expresión regular:  $((y|z)|x(y|z))^* (\epsilon | x)$



- Cadenas que pertenecen al lenguaje:
  - $w_1 = yxz$
  - $w_2 = zxyxz$
  - $w_3 = zzzxzzz$
  - $w_4 = yxzx$
  - $w_5 = zxz$
- Cadenas que no pertenecen al lenguaje:
  - $w_6 = xx$
  - $w_7 = zyxxxy$
  - $w_8 = zyyzxx$
  - $w_9 = xxyzx$
  - $w_{10} = zxxz$

## 2. Ejercicios sobre operadores extendidos

### 2.1. Direcciones de correos electrónicos de estudiantes de la Universidad de La Laguna.

- Expresión regular: `^alu\d{10}@ull\.edu\.es$`

REGULAR EXPRESSION      5 matches (135 steps, 60µs)      ⓘ

`^alu\d{10}@ull\.edu\.es$`      / gm      ⚏

TEST STRING

```
alu0101317038@ull.edu.esd
alu0123456789@ull.edu.esd
alu0475821452@ull.edu.esd
alu5723839456@ull.edu.esd
alu1112223334@ull.edu.esd
alu0101317038@ull.edu.essd
aalu0101317038@ull.edu.esd
alu03@ull.edu.esd
alu0101317038@ull.edu.comd
alurtrtrrt@ull.edu.esd
```

## 2.2. Palabras que terminen por una vocal.

- Expresión regular:  $^*[a-zA-Z]*[aeiouAEIOU]$$

REGULAR EXPRESSION      5 matches (61 steps, 60µs)      ⓘ

: /  $^*[a-zA-Z]*[aeiouAEIOU]$$       / gm      ⚒

TEST STRING

```
Calamardo1
pistola1
AMERICA1
cartera1
mano1
ordenador1
papel1
```

## 2.3. Números enteros.

- Expresión regular:  $^*[+ -]?\d+$$

REGULAR EXPRESSION      5 matches (38 steps, 45µs)      ⓘ

: /  $^*[+ -]?\d+$$       / gm      ⚒

TEST STRING

```
-21
01
11
+21
-341
4.51
-22.31
```

## 2.4. Texto que se encuentre entre paréntesis.

- Expresión regular: `\(([^\(\)]*)\)`

REGULAR EXPRESSION

5 matches (20 steps, 40µs) / gm

TEST STRING

```
: / \(([^\(\)]*)\)
(hola) ↵
hola•que•tal•(me•llamo•alejandro) ↵
(123) ↵
(estoy•es•una ↵
prueba) ↵
↵
(####@) ↵
↵
soto ↵
musica ↵
canciones) ↵
(probando ↵
```

## 2.5. Código postal en España.

- Expresión regular: `^(0[1-9]|1-4)\d|5[0-2])\d{3}$`

The screenshot shows a regular expression tester interface. At the top, it says "REGULAR EXPRESSION" and "5 matches (61 steps, 50μs)". The regular expression input field contains: `^(0[1-9]|1-4)\d|5[0-2])\d{3}$`. Below that is a "TEST STRING" input field containing:  
00000  
01000  
22456  
38340  
59888  
52340  
45333

## 2.6. Palabras que contienen sólo letras mayúsculas.

- Expresión regular: `^[A-ZÁÉÍÓÚÜÑ]+$`

The screenshot shows a regular expression tester interface. At the top, it says "REGULAR EXPRESSION" and "5 matches (27 steps, 35μs)". The regular expression input field contains: `^[A-ZÁÉÍÓÚÜÑ]+$`. Below that is a "TEST STRING" input field containing:  
CARACOLA  
caMión  
MÓVIL  
GRUÑÓN  
DIPLOMA  
TECLADO

2.7. Números de teléfono en formato prefijo XXX-XXX-XXX, donde el prefijo del país puede indicarse empezando por 00 o bien con un símbolo +; por ejemplo, 0034 o +34 para España.

- Expresión regular: `^(00|+)\d{1,3}\s\d{3}-\d{3}-\d{3}$`

REGULAR EXPRESSION      8 matches (119 steps, 75µs)      ⓘ

```
:/ ^(00|+)\d{1,3}\s\d{3}-\d{3}-\d{3}$ / gm
```

TEST STRING

```
+34 600-123-456d
0034 987-654-321d
+1 123-456-789d
001 555-123-456d
+44 020-123-456d
0049 030-123-456d
+81 123-456-789d
00358 123-456-789
```

## 2.8. Fecha en formato DD/MM/AAAA

- Expresión regular: `^(0[1-9]|1[2][0-9]|3[01])/(0[1-9]|1[0-2])/[0-9]{4}$`

REGULAR EXPRESSION      6 matches (97 steps, 60µs)      ⓘ

```
:/ ^(0[1-9]|1[2][0-9]|3[01])/(0[1-9]|1[0-2])/[0-9]{4}$ / gm
```

TEST STRING

```
01/01/0000d
09/09/1999d
10/10/2020d
31/12/2025d
30/04/2024d
28/02/2023
```

## 2.9. Palabras de al menos 10 letras de longitud.

- Expresión regular: `^([A-Za-zÁÉÍÓÚÜÑáéíóúüñ]{10,})$`

REGULAR EXPRESSION

`^([A-Za-zÁÉÍÓÚÜÑáéíóúüñ]{10,})$` / gm

TEST STRING

```
programación
construcción
verificación
montañismo
língüística
rápidamente
```

6 matches (29 steps, 50µs)

This screenshot shows a regular expression testing interface. The regular expression input field contains `^([A-Za-zÁÉÍÓÚÜÑáéíóúüñ]{10,})$`. The test string area contains several Spanish words: "programación", "construcción", "verificación", "montañismo", "língüística", and "rápidamente". The results section indicates 6 matches found in 29 steps, with a total execution time of 50µs. The matches are highlighted in blue in the test string list.

## 2.10. Palabras que terminen con “ing” o “ed”.

- Expresión regular: `(?i)^([A-Za-zÁÉÍÓÚÜÑáéíóúüñ]+(ing|ed)$)`

REGULAR EXPRESSION

`(?i)^([A-Za-zÁÉÍÓÚÜÑáéíóúüñ]+(ing|ed)$)` / gm

TEST STRING

```
played
making
dancing
wasted
coche
ordenador
CODING
```

5 matches (144 steps, 65µs)

This screenshot shows a regular expression testing interface. The regular expression input field contains `(?i)^([A-Za-zÁÉÍÓÚÜÑáéíóúüñ]+(ing|ed)$)`. The test string area contains several words: "played", "making", "dancing", "wasted", "coche", "ordenador", and "CODING". The results section indicates 5 matches found in 144 steps, with a total execution time of 65µs. The matches are highlighted in green in the test string list.

## 3. Modificación

### 3.1. Sentencias class en C++

- Expresión regular:

```
^class\s[A-Z]\w*\s(\:(public|protected|private)\s[A-Z]\w+(\.,)*\s\{\D*\}\};$
```

- Cadenas que si acepta:

REGULAR EXPRESSION

```
:/ ^class\s[A-Z]\w*\s(\:(public|protected|private)\s[A-Z]\w+(\.,)*\s\{\D*\}\};$ / gm
```

TEST STRING

```
class=Gato:public=Animal={hola};  
class=Gato={hola};  
class=Perro:protected=Animal,=public=Mamifero={hola};  
class=Cocher:private=Vehiculo={};  
class=Moto{};
```

- Cadenas que no acepta:

REGULAR EXPRESSION

```
:/ ^class\s[A-Z]\w*\s(\:(public|protected|private)\s[A-Z]\w+(\.,)*\s\{\D*\}\};$ / gm
```

TEST STRING

```
class=hola;  
class=Moto  
hola=mundo  
class=COCCHE;  
class=Perro{},
```

### 3.2. Lenguaje binario con número par de 1's

- Expresión regular:  $(0|0^*10^*10^*)^*$

- Cadenas que si acepta:

A screenshot of a terminal window. The command entered is `: / (0|0*10*10*)*`. The output shows a single match for the string "01110110101010111111001111111". The match is highlighted with blue boxes around the segments "0", "10101010", and "111111100", and green boxes around the segments "011", "110110101", and "11111111". The terminal also displays the count "1 matches (0 to 69 bytes)" and the option "/ gm".

- Cadenas que no acepta:

A screenshot of a terminal window. The command entered is `: / (0|0*10*10*)*`. The output shows a single match for the string "1111010101011111001111111". The match is highlighted with blue boxes around the segments "10101010", "111111100", and "11111111", and green boxes around the segments "111", "110110101", and "11111111". The terminal also displays the count "1 matches (0 to 69 bytes)" and the option "/ gm".