

# ANÁLISIS COMPARATIVO DE LENGUAJES

Guía de estudio correspondiente a la Teoría sobre: Aspectos Formales de los Lenguajes de Programación



# ASPECTOS FORMALES DE LOS LENGUAJES DE PROGRAMACIÓN PRIMERA PARTE

*Notas de Clase*

*Capítulos III – Programming Languages – Design and Implementation –  
Terrence Pratt*

*Capítulos III Concepts of Programming Languages – Robert Sebesta*

*Apunte de la cátedra.*

# OBJETIVOS DE ESTA TEORÍA

- Introducir métodos formales para la especificación de lenguajes programación.
- Definiciones necesarias para comenzar el estudio de modelos formales.
- Clasificación de lenguajes: Jerarquía de Chomsky.
- Expresiones Regulares.

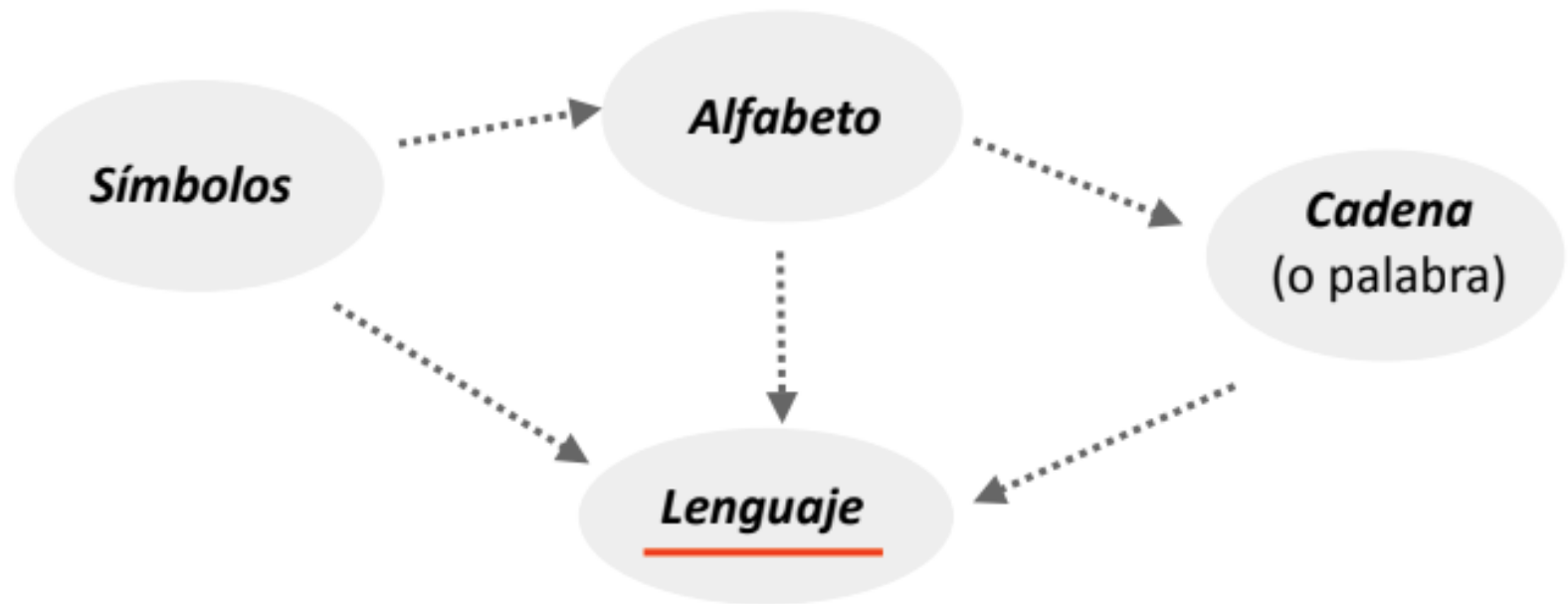
# MODELOS PARA LA DESCRIPCIÓN FORMAL DE LENGUAJES DE PROGRAMACIÓN

- A partir de ahora es nuestro objetivo estudiar métodos que permitan la descripción formal de lenguajes, es decir el cómo describir sintaxis.
- **Chomsky** (1950, 1959). Lingüista, estudió la naturaleza teórica de los lenguajes naturales.
- En particular nuestro interés es estudiar cómo se definen los lenguajes de programación.
- Para comenzar debemos dar algunas **definiciones previas** necesarias para poder definir modelos formales:



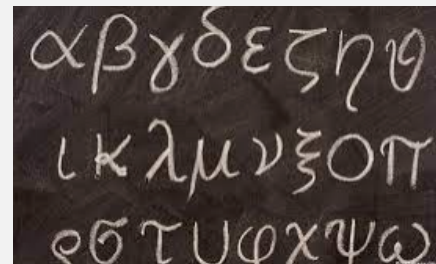
# DEFINICIONES Y CONCEPTOS INTRODUCTORIOS

## *CADENAS, ALFABETOS Y LENGUAJES*



# DEFINICIONES Y CONCEPTOS INTRODUCTORIOS *SÍMBOLO*

Un **símbolo** es una entidad abstracta a la que no definiremos formalmente, como el punto en geometría. Letras y dígitos son ejemplos de símbolos frecuentemente usados.



# DEFINICIONES Y CONCEPTOS INTRODUCTORIOS ALFABETOS

un *alfabeto* es un conjunto finito, no vacío de símbolos. Se utiliza comúnmente el símbolo  $\Sigma$ , para denotarlo.

$$\Sigma_1 = \{a,b\}$$

$$\Sigma_2 = \{0,1\}$$

$$\Sigma_4 = \{0,1,2,\dots,9\}$$

$$\Sigma_3 = \{a,b,c,d,\dots,z\}$$

ASCII



0000	0	0001	@	0040	P	0080	`	00C0	
0001	1	0002	A	0041	Q	0081	a	00C1	
0002	2	0003	B	0042	R	0082	b	00C2	
0003	3	0004	C	0043	S	0083	c	00C3	
0004	4	0005	D	0044	T	0084	d	00C4	
0005	5	0006	E	0045	U	0085	e	00C5	
0006	6	0007	F	0046	V	0086	f	00C6	
0007	7	0008	G	0047	W	0087	g	00C7	
0008	8	0009	H	0048	X	0088	h	00C8	
0009	9	0010	I	0049	Y	0089	i	00C9	
0010		0011	J	0050	Z	0090	j	00CA	
0011		0012	K	0051	[	0091	k	00CB	
0012		0013	L	0052	\	0092	l	00CC	
0013		0014	M	0053	]	0093	m	00CD	
0014		0015	N	0054	^	0094	n	00CE	
0015		0016	O	0055	_	0095	o	00CF	
0016		0017	P	0056	`	0096	p	00D0	
0017		0018	Q	0057	{	0097	q	00D1	
0018		0019	R	0058		0098	r	00D2	
0019		0020	S	0059	}	0099	s	00D3	
0020		0021	T	0060	~	00A0		00D4	
0021		0022	U	0061		00A1		00D5	
0022		0023	V	0062		00A2		00D6	
0023		0024	W	0063		00A3		00D7	
0024		0025	X	0064		00A4		00D8	
0025		0026	Y	0065		00A5		00D9	
0026		0027	Z	0066		00A6		00DA	
0027		0028	[	0067		00A7		00DB	
0028		0029	\	0068		00A8		00DC	
0029		0030	]	0069		00A9		00DD	
0030		0031	^	0070		00AA		00DE	
0031		0032	_	0071		00AB		00DF	
0032		0033	`	0072		00AC		00E0	
0033		0034	{	0073		00AD		00E1	
0034		0035		0074		00AE		00E2	
0035		0036	}	0075		00AF		00E3	
0036		0037	~	0076		00B0		00E4	
0037		0038		0077		00B1		00E5	
0038		0039		0078		00B2		00E6	
0039		0040		0079		00B3		00E7	
0040		0041		0080		00B4		00E8	
0041		0042		0081		00B5		00E9	
0042		0043		0082		00B6		00EA	
0043		0044		0083		00B7		00EB	
0044		0045		0084		00B8		00EC	
0045		0046		0085		00B9		00ED	
0046		0047		0086		00BA		00EE	
0047		0048		0087		00BB		00EF	
0048		0049		0088		00BC		00F0	
0049		0050		0089		00BD		00F1	
0050		0051		0090		00BE		00F2	
0051		0052		0091		00BF		00F3	
0052		0053		0092		00C0		00F4	
0053		0054		0093		00C1		00F5	
0054		0055		0094		00C2		00F6	
0055		0056		0095		00C3		00F7	
0056		0057		0096		00C4		00F8	
0057		0058		0097		00C5		00F9	
0058		0059		0098		00C6		00FA	
0059		0060		0099		00C7		00FB	
0060		0061		0100		00C8		00FC	
0061		0062		0101		00C9		00FD	
0062		0063		0102		00CA		00FE	
0063		0064		0103		00CB		00FF	
0064		0065		0104		00CC			
0065		0066		0105		00CD			
0066		0067		0106		00CE			
0067		0068		0107		00CF			
0068		0069		0108		00D0			
0069		0070		0109		00D1			
0070		0071		0110		00D2			
0071		0072		0111		00D3			
0072		0073		0112		00D4			
0073		0074		0113		00D5			
0074		0075		0114		00D6			
0075		0076		0115		00D7			
0076		0077		0116		00D8			
0077		0078		0117		00D9			
0078		0079		0118		00DA			
0079		0080		0119		00DB			
0080		0081		0120		00DC			
0081		0082		0121		00DD			
0082		0083		0122		00DE			
0083		0084		0123		00DF			
0084		0085		0124		00E0			
0085		0086		0125		00E1			
0086		0087		0126		00E2			
0087		0088		0127		00E3			
0088		0089		0128		00E4			
0089		0090		0129		00E5			
0090		0091		0130		00E6			
0091		0092		0131		00E7			
0092		0093		0132		00E8			
0093		0094		0133		00E9			
0094		0095		0134		00EA			
0095		0096		0135		00EB			
0096		0097		0136		00EC			
0097		0098		0137		00ED			
0098		0099		0138		00EE			
0099		0100		0139		00EF			
0100		0101		0140		00F0			
0101		0102		0141		00F1			
0102		0103		0142		00F2			
0103		0104		0143		00F3			
0104		0105		0144		00F4			
0105		0106		0145		00F5			
0106		0107		0146		00F6			
0107		0108		0147		00F7			
0108		0109		0148		00F8			
0109		0110		0149		00F9			
0110		0111		0150		00FA			
0111		0112		0151		00FB			
0112		0113		0152		00FC			
0113		0114		0153		00FD			
0114		0115		0154		00FE			
0115		0116		0155		00FF			
0116		0117		0156					
0117		0118		0157					
0118		0119		0158					
0119		0120		0159					
0120		0121		0160					
0121		0122		0161					
0122		0123		0162					
0123		0124		0163					
0124		0125		0164					
0125		0126		0165					
0126		0127		0166					
0127		0128		0167					
0128		0129		0168					
0129		0130		0169					
0130		0131		0170					
0131		0132		0171					
0132		0133		0172					
0133		0134		0173					
0134		0135		0174					
0135		0136		0175					
0136		0137		0176					
0137		0138		0177					
0138		0139		0178					
0139		0140		0179					
0140		0141		0180					
0141		0142		0181					
0142		0143		0182					
0143		0144		0183					
0144		0145		0184					
0145		0146		0185					
0146		0147		0186					
0147		0148		0187					
0148		0149		0188					
0149		0150		0189					
0150		0151		0190					
0151		0152		0191					
0152		0153		0192					
0153		0154		0193					
0154		0155		0194					
0155		0156		0195					
0156		0157		0196					
0157		0158		0197					
0158		0159		0198					
0159		0160		0199					
0160		0161		0200					
0161		0162		0201					
0162		0163		0202					
0163		0164		0203					
0164		0165		0204					
0165		0166		0205					
0166		0167		0206					
0167		0168		0207					
0168		0169		0208					
0169		0170		0209					
0170		0171		0210					
0171		0172		0211					
0172		0173		0212					
0173		0174		0213					
0174		0175		0214					
0175		0176		0215					
0176		0177		0216					
0177		0178		0217					
0178		0179		0218					
0179		0180		0219					
0180		0181		0220					
0181		0182		0221					
0182		0183		0222					
0183		0184		0223					
0184		0185		0224					
0185		0186		0225		</			

# DEFINICIONES Y CONCEPTOS INTRODUCTORIOS *CADENAS*

Una **cadena** o **palabra** es una secuencia finita de símbolos.  
Por ejemplo,  $a$ ,  $b$  y  $c$  son símbolos pertenecientes a un alfabeto  
y  $cccab$  es una cadena.



### Ejemplos

$$\Sigma_1 = \{a, b\}$$

$$x_1 = \text{aabbbaa}; \quad x_2 = \text{bbbaa} ; \dots$$

$$\Sigma_2 = \{0,1\}$$

$$x_1 = 1011011; \quad x_2 = 1100; \dots$$



# DEFINICIONES Y CONCEPTOS INTRODUCTORIOS CADENAS

## Otros ejemplos de cadenas

```

package ifelse;
import java.util.Scanner;

public class cifra {
    int num;
    Scanner sc=new Scanner(System.in);
    System.out.println("Ingrese numero");
    num=sc.nextInt();
    if(num>0&&num<10){
        System.out.println("El numero es de un
    }else{
        if(num>=10&&num<100){
            System.out.println("El numero es de un
        }else{
            if(num>=100&&num<1000){
                System.out.println("El numero es
            }else{
                if(num>=1000&&num<10000){
                    System.out.println("El numero es d
                }
            }
        }
    }
}

```

```

#include <stdio.h>
#include <stdlib.h>

//declaramos la funcion suma, que recibe dos enteros
void suma(int n1, int n2)
{
    int resultado; //para guardar el resultado de la suma
    resultado = n1 + n2; //asigno el resultado de la suma a la var
    printf("La suma de ambos numeros es: %i\n", resultado);
}

int main()
{
    int num1, num2; //creamos dos enteros, num1 y
    //damos valores a estas variables
    num1 = 3;
    num2 = 7;
    //llamada a la función "Suma"
    suma(num1, num2);
    system("PAUSE");
}

```

$\Sigma =$

0000	0	0030	@	0040	P	0050	`	0060	p	0070	0080	°	0090	À	00C0	Ð
0001	1	0031	A	0041	Q	0051	a	0061	q	0071	j	0081	±	0091	Á	Ñ
0002	2	0032	B	0042	R	0052	b	0062	r	0072	l	0082	²	0092	Â	Ò
0003	3	0033	C	0043	S	0053	c	0063	s	0073	t	0083	³	0093	Ã	Ó
0004	4	0034	D	0044	T	0054	d	0064	t	0074	u	0084	´	0094	Ä	Ô
0005	5	0035	E	0045	U	0055	e	0065	u	0075	v	0085	µ	0095	Å	Õ
0006	6	0036	F	0046	V	0056	f	0066	v	0076	w	0086	¶	0096	Æ	Ö
0007	7	0037	G	0047	W	0057	g	0067	w	0077	x	0087	·	0097	Ç	×

```

<html>
<head>
<title>Contact us
<meta http-equiv="
<link rel="short
<script type="t
ead> class

```

# DEFINICIONES Y CONCEPTOS INTRODUCTORIOS *CADENAS*

- ***Cadena vacía***

Es la cadena con 0 ocurrencias de símbolos y denotada por la letra griega  $\lambda$   
(sin importar el alfabeto)

## *Ejemplos*

$$\Sigma_1 = \{a, b\}$$

$$x = \lambda$$

$$\Sigma_2 = \{0, 1\}$$

$$x = \lambda$$

# DEFINICIONES Y CONCEPTOS INTRODUCTORIOS CADENAS

## Operaciones sobre cadenas

Longitud de una cadena denotada por  $|x|$

Cantidad de símbolos de la cadena  $x$

$$x = a_1 a_2 a_3 \dots a_n \quad \longrightarrow \quad |x| = |a_1 a_2 a_3 \dots a_n| = n$$

*Ejemplo*

$$|x| = |1011011| = 7$$

$$|x| = |123+32*339+57/65| = 16$$

# DEFINICIONES Y CONCEPTOS INTRODUCTORIOS *CADENAS*

## *Operaciones sobre cadenas*

*Longitud de una cadena: cadena nula*

$$x = \lambda \quad \Rightarrow \quad |x| = |\lambda| = 0$$

# DEFINICIONES Y CONCEPTOS INTRODUCTORIOS CADENAS

## Operaciones sobre cadenas

Concatenación de cadenas:

Es una operación binaria que permite concatenar cadenas  
(se colocan los símbolos de una cadena  $y$  a continuación de  
los símbolos de la cadena  $x$ )

$$\begin{array}{lcl} x = a_1 a_2 a_3 \dots a_n & \Rightarrow & x y = a_1 a_2 a_3 \dots a_n b_1 b_2 b_3 \dots b_m \\ y = b_1 b_2 b_3 \dots b_m & & |x y| = n + m \end{array}$$

La cadena vacía ( $\lambda$ ) es el neutro de la concatenación

$$\lambda x = x \lambda = x$$

# DEFINICIONES Y CONCEPTOS INTRODUCTORIOS CADENAS

## Operaciones sobre cadenas

Potencia sobre cadenas:

Potencia sobre Cadenas  $\Rightarrow x^n$

$$x^0 = \lambda$$

$$x^n = x^{n-1} x$$

Ejemplos:

Si  $x = abaa$  entonces:

$$x^2 = abaaabaa$$

$$x^3 = abaaabaaabaa$$

¿Cómo se obtiene  $x^3$  según la definición?

# DEFINICIONES Y CONCEPTOS INTRODUCTORIOS CADENAS

## Operaciones sobre cadenas

### Potencia sobre Alfabetos $\Sigma^n$

Es el conjunto de todas las cadenas de longitud  $n$  cuyos símbolos están en  $\Sigma$ .

$$\Sigma^0 = \{ \lambda \}$$

$$\Sigma^n = \{ x_1, x_2, \dots, x_i, \dots \} \quad \text{donde } \forall x_i \quad |x_i| = n$$

### Ejemplos:

1. Notar que  $\Sigma^0 = \{ \lambda \}$ , independientemente de lo que sea  $\Sigma$ .  
Dado que  $\lambda$  es la única cadena cuya longitud es 0.
2. Si  $\Sigma = \{0, 1\}$ , entonces  $\Sigma^1 = \{0, 1\}$ ,
3.  $\Sigma^2 = \{00, 01, 10, 11\}$ ,
4.  $\Sigma^3 = \{000, 001, 010, 011, 100, 101, 110, 111\}$  y así siguiendo.

# DEFINICIONES Y CONCEPTOS INTRODUCTORIOS *LENGUAJES*

Un lenguaje  $L$  es un conjunto (finito o infinito) de cadenas sobre algún alfabeto  $\Sigma$

## *Ejemplos*

$$\Sigma_1 = \{a, b\}$$

$$L_1 = \emptyset ; L_2 = \{aabbbaa, bbaa, bb, b, bbbbaaabb, \dots\}; \dots$$

$$\Sigma_2 = \{0, 1\}$$

$$L_1 = \{0, 1\}; L_2 = \{1001, 100, 1, 000, 11111, \dots\}; \dots$$

**Observación:** el lenguaje  $L$  no necesariamente debe incluir todos los símbolos del alfabeto



# DEFINICIONES Y CONCEPTOS INTRODUCTORIOS *LENGUAJES*

Otros ejemplos de Lenguajes



# DEFINICIONES Y CONCEPTOS INTRODUCTORIOS *LENGUAJES*

## *Lenguaje Referencial*

Es el conjunto de todas las cadenas sobre algún alfabeto  $\Sigma$   
y es denotado por  $\Sigma^*$

$$\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \dots = \{\lambda\} \cup \Sigma^1 \cup \Sigma^2 \cup \dots = \bigcup_{n \in (\mathbb{N} \cup \{0\})} \Sigma^n$$

Ejemplo:

$$\Sigma = \{0,1\} \quad \Sigma^* = \{\lambda, 0, 1, 00, 11, 010, 000, \dots\}$$

# DEFINICIONES Y CONCEPTOS INTRODUCTORIOS *LENGUAJES*

- $\Sigma^*$  es un conjunto infinito de cadenas de longitud finita, ya que  $\Sigma$  es un conjunto finito no vacío de símbolos.
- Para cualquier lenguaje  $L$  se cumple que  $L \subseteq \Sigma^*$
- Y además:

$$\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \dots = \bigcup_{n \in \mathbb{N}} \Sigma^n$$

$$\Sigma^+ = \Sigma^* - \Sigma^0$$

# DEFINICIONES Y CONCEPTOS INTRODUCTORIOS *LENGUAJES*

## Formas de describirlos

Primera opción: descripción del conjunto por extensión



$$L = \{x_1, x_2, x_3, x_4, \dots\}$$

**Ejemplo:**

$$\Sigma = \{b, c\}$$

$$L = \{c, cc, ccc, cccc\}$$

# DEFINICIONES Y CONCEPTOS INTRODUCTORIOS LENGUAJES

## Formas de describirlos

Segunda opción: descripción del conjunto por comprensión



$$L = \{x \in \Sigma^* / \mathcal{P}(x)\}$$

Ejemplo:

$$\Sigma = \{a, b\}$$

$$L = \{x \in \Sigma^* / \underbrace{x \text{ contiene una cantidad par del símbolo } a \text{ y} \\ \text{no contiene el símbolo } b}_{\mathcal{P}(x)}\}$$

# DEFINICIONES Y CONCEPTOS INTRODUCTORIOS LENGUAJES

## Formas de describirlos

**Otra opción:** remplazar  $x$  por una expresión con parámetros y describir el lenguaje estableciendo las condiciones sobre los parámetros.

Ejemplo

$\Sigma = \{a, b\}$

$L = \{a^n / n \in \mathbb{N} \text{ y } n \text{ es par}\}$

(expresión, donde  $n$  es el parámetro)

(condición sobre el parámetro  $n$ )

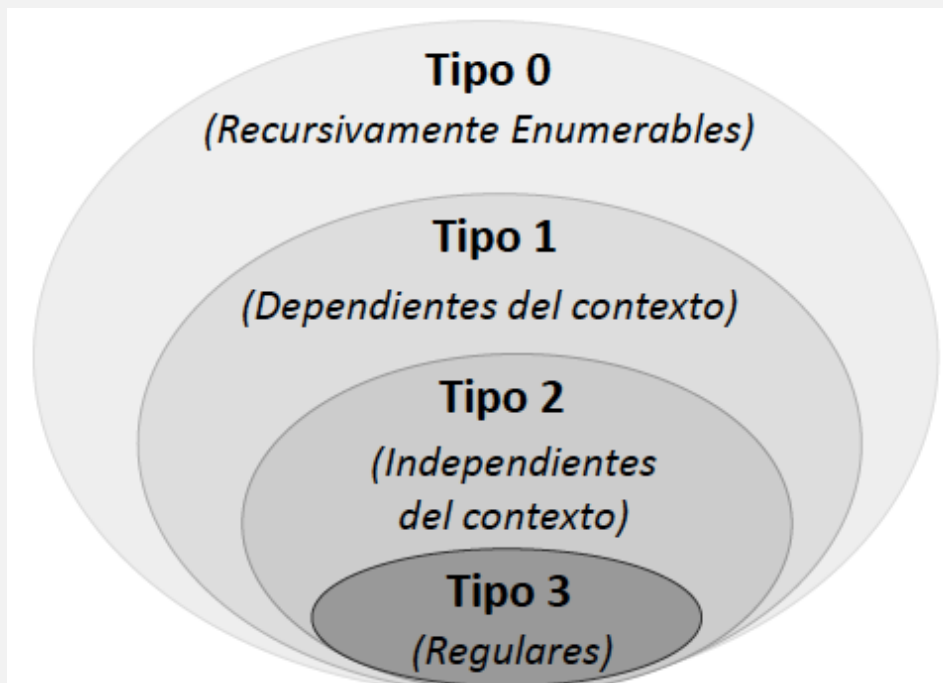
**Observación:** también es posible describir un lenguaje, utilizando lenguaje natural.

# LENGUAJES FORMALES

## Jerarquía de Chomsky

Modelos descriptores de Lenguajes:

- ✓ Dispositivos reconocedores: Autómatas o Máquinas abstractas o Máquinas de Estados
- ✓ Dispositivos generadores: Gramáticas



Noam Chomsky

# DISPOSITIVOS DESCRIPTORES DE LENGUAJES

**Gramáticas:** es un modelo matemático que permite generar a través de reglas sintácticas o gramaticales, cadenas miembros de un lenguaje específico.

**Autómatas:** es un modelo matemático que representa la idea de computación o manipulación de cadenas vía la aplicación de acciones preestablecidas. Tiene como objetivo (en general), determinar la pertenencia de una cadena a un lenguaje específico.



# LENGUAJES REGULARES



También se puede  
escribir como

$$L = \{x \in \{0,1\}^* / x \text{ comienza con } 0\}$$

Lenguajes Regulares



$$\Sigma = \{0, 1\}$$

$$L_1 = \{\lambda, 00, 11\}$$

$$L_2 = \{0^i 1 / 1 \leq i \leq 5\}$$

Finitos

$$L_3 = \{x \in \Sigma^* / x \text{ comienza con } 0\}$$

$$L_4 = \{1^k / k > 0\}$$

$$L_5 = \{0^i 1^k / i \geq 1 \text{ y } k > 0\}$$

Infinitos

# LENGUAJES REGULARES

➤ Otros ejemplos de lenguajes regulares son:

Los nombres de los identificadores, palabras claves, números, etc., en los lenguajes de programación

# EXPRESIONES REGULARES (ER)

- Las **ER** son un tipo de notación particular que permite describir lenguajes regulares.
- Ofrecen una forma declarativa de expresar las cadenas.
- Se construyen utilizando los siguientes elementos:

## ■ operandos:

*los símbolos  $\sigma$  en  $\Sigma$*   
*la palabra vacía ( $\lambda$ )*  
*el conjunto vacío ( $\emptyset$ )*  
*expresiones regulares*

## ■ operadores:

*unión (+)*  
*concatenación (.)*  
*clausura de Kleene (\*)*

# EXPRESIONES REGULARES (ER)

- La ER se definen recursivamente de la siguiente manera:

$\emptyset$  es una ER y denota el conjunto vacío,  $L(\emptyset) = \emptyset$ .

$\lambda$  es una ER y denota el conjunto  $\{\lambda\}$ .  $L(\lambda) = \{\lambda\}$ .

$\forall a \in \Sigma$ ,  $a$  es una ER y denota el conjunto  $\{a\}$ .  $L(a) = \{a\}$ .

Si  $\alpha$  y  $\beta$  son ER, entonces:

$\alpha + \beta$  es una ER y denota el conjunto  $L(\alpha + \beta) = L(\alpha) \cup L(\beta)$

$\alpha\beta$  es una ER y denota el conjunto  $L(\alpha\beta) = L(\alpha)L(\beta)$

$\alpha^*$  es una ER y denota el conjunto  $L(\alpha^*) = (L(\alpha))^*$

$(\alpha)$  es una ER y denota el conjunto  $L((\alpha)) = L(\alpha)$

Ninguna otra cosa es una ER.

# EXPRESIONES REGULARES (ER)

La **Precedencia** de los operadores (de mayor a menor): clausura, concatenación y unión. La precedencia puede alterarse usando paréntesis.

La **Asociatividad**: de izquierda a derecha.

## Ejemplos:

Considerando  $\Sigma = \{a, b\}$

- $a$  es una ER que denota el lenguaje  $L = \{a\}$
- $b$  es una ER que denota el lenguaje  $L = \{b\}$
- $a + b$  es una ER que denota el lenguaje  $L = \{a\} \cup \{b\} = \{a, b\}$
- $ab$  es una ER que denota el lenguaje  $L = \{a\} \cdot \{b\} = \{ab\}$
- $(ab)^*$  es una ER que denota el lenguaje  $L = \{ab\}^*$
- $(a + b)(ab)^*$  es una ER que denota el lenguaje  $L = \{a, b\}\{ab\}^*$

## EXPRESIONES REGULARES (ER)

Construir una ER para cada uno de los siguientes lenguajes, considerando  $\Sigma = \{a, b\}$ :

➤  $\{\lambda, aab, baa\}$

➤  $L = \{x \in \Sigma^* / x \text{ comienza con } b \text{ y termina con } a\}$

➤  $L = \{x \in \Sigma^* / x \text{ contiene la subcadena } ab\}$

➤ El lenguaje de todas las cadenas que tienen número par de símbolos.

## EXPRESIONES REGULARES (ER)

- $(\lambda + aab + baa)$
- $b(a + b)^*a$
- $(a + b)^*ab(a + b)^*$
- $(aa + ab + ba + bb)^*$

*Dar ejemplos de cadenas pertenecientes al lenguaje denotado por cada una de las ER. (desarrollado en clase)*

## OTROS USOS DE LAS ER

- Describir patrones para búsqueda en textos (palabras terminadas en aba, números telefónicos, direcciones de mail, etc., etc.).
- *Generar Analizadores Lexicográficos.*
- Comandos de búsqueda en SO como Linux (ejemplo: `$ ls -l *.ps | grep May`).
- Etc.



¿DUDAS?

