

# **Cuaderno de Prácticas**

## **Modelos de Computación**

### **Grupo B**

Miguel Santiago Cervilla 76438234B

# Práctica 1:

## Práctica 1 Modelos Computación

(1)  $H = (N, \Sigma, P, S)$  donde el alfabeto es  $\Sigma = \{a, b\}$

Dado esta gramática demuestra que se genera el siguiente lenguaje:  $L(H) = \{u^* | u \in \{ab\}^*\}$  y  $P_a(u) = P_b(u)$

Símbolos terminales:  $\{a, b\}$

Reglas de producción:  $A \rightarrow a$ ,  $A \rightarrow aS$ ,  $A \rightarrow bA$

$\vdash aA \Rightarrow aB \Rightarrow b \quad B \Rightarrow bS \Rightarrow aBB \quad \text{y} \quad aA \Rightarrow bA \Rightarrow bB \Rightarrow bB \Rightarrow b$

Símbolo de partida:  $S \rightarrow aB$ ,  $S \rightarrow bA$

1º Demuestra que el generador sólo genera cadenas que verifican  $\leftarrow L$

$P_a(u) = P_b(u)$ ,  $(N, H) \models u \vdash \leftarrow L$

Para demostrarlo, nos vemos en la figura en las producciones de A y B. Fijándonos en las producciones de A, se observa claramente que lo que hace es añadir una a más que b y fijándonos en las producciones de B cuantos lo contara, se añade una b más que a. Dependiendo de la cláusula inicial que usemos, se añade una a o una b.

$$S \rightarrow aB \quad S \rightarrow bA$$

Con estas dos producciones símbolos de partida, nos vamos a asegurar que se cumpla que el nº de a y b sea el mismo ya que al inicio se introduce una a o b para que las siguientes producciones hagan que el nº de a y b sea el mismo.

29 Demostrar que  $\sigma$  genera todos los estados  $P_a(u) = P_b(u)$

Con lo visto en el punto anterior, sabiendo que las producciones de  $A$  añaden una  $a$  de más y que las de  $B$  añaden una  $B$  de más, podemos deducir que siempre se va a cumplir que  $P_a(u) = P_b(u)$ .

Para ello vamos a comprobar algunos ejemplos:

$$P_a(u) = u = P_b(u) \quad \text{dado}$$

$$\begin{aligned} S &\rightarrow aB \rightarrow aaBB \rightarrow \cancel{aaa}B\cancel{aa}BB \rightarrow aaaBBaBB \rightarrow \\ &\rightarrow aaabbabb; \quad P_a(u) = u = P_b(u) \end{aligned}$$

$$S \rightarrow bA \rightarrow ba \quad P_a(u) = 1 = P_b(u)$$

## Práctica 2:

Práctica 2

Comprobar que la siguiente gramática (de tipo 2) genera un lenguaje de tipo 3.

$$G = \{ S \mid A, B \} , \{ a, b, c, d \}, P, S \}$$

Reglas producción:

$$S \rightarrow AB \quad A \rightarrow Ab \quad A \rightarrow a$$

$$B \rightarrow cB \quad B \rightarrow d$$

$$a, b, c, d \in \mathbb{N}$$

Cómo podemos observar, claramente se trata de una gramática de tipo 2. ( $S \rightarrow AB$ )

Primero vamos a comprobar que claramente cumple ese lenguaje. Para ello usamos las reglas de producción para ver si genera este lenguaje.

$$S \rightarrow AB \rightarrow AbB \rightarrow abcB \rightarrow abcd$$

$$S \rightarrow AB \rightarrow ad$$

$$S \rightarrow AB \rightarrow abbcccd$$

Cómo se puede observar a simple vista mirando las reglas de producción, puedes observar que siempre se generará cadenas del tipo  $a^i b^j c^k d^l$   $i, j, k, l \in \mathbb{N}$ .

i, j pueden ser O o y la gramática

Siempre se va a tener a al principio y va d  
al final de la cadena.

Una vez comprobado que se cumple, vamos a intentar buscar una gramática de tipo 3 que lo cumpla.

Para ello debemos realizar una serie de transformaciones en las reglas de producción que tenemos las cuales son:

$$S \rightarrow AB \quad A \rightarrow Ab \quad A \rightarrow a \quad B \rightarrow cB \quad B \rightarrow d$$

Para que sean del tipo 3 deben ser de estos tipos:

$$S \rightarrow aB \quad S \rightarrow A \quad S \rightarrow a$$

Muestra sentencia inicial, no lo cumple, para ella te modificas con una que si lo cumple.

$$S \rightarrow aB \quad (\text{así aseguramos que siempre saldrá la } a \text{ primero})$$

Ahora debemos buscar cláusulas que generen todas las b necesarias:

$$B \rightarrow bB$$

Una vez generadas las b, necesitamos generar las C

$$B \rightarrow C \quad C \rightarrow CC$$

Así generaremos todas las C y parece justificar debemos acabar con una d:

$$C \rightarrow d$$

Una vez presentadas estas conjuntos de cláusulas, podemos concluir que este lenguaje puede ser sintetizado con un lenguaje lenguaje puede ser resuelto con una gramática de tipo 3, y es el siguiente:

$$S \rightarrow aB \quad B \rightarrow bB \quad B \rightarrow C \quad C \rightarrow cC \quad C \rightarrow d$$

comprendiendo que las reglas escritas en la parte superior del diagrama están escritas en la parte inferior.

$$b = B \quad B = d \quad a = A \quad dA = A \quad dB = C$$

donde tanto  $B$  como  $C$  y  $A$  son tipos de variables.

$$\therefore A \leftarrow 2 \quad B \leftarrow 2 \quad C \leftarrow 2$$

(señalando el valor inicial, el punto de inicio, los tipos de variables y el valor final de cada variable).

$$(señalando el valor inicial, el punto de inicio, los tipos de variables y el valor final de cada variable).$$

Al finalizar el cálculo se obtiene el resultado:

$$B \rightarrow P_B$$

o sea, una condición para el cálculo.

$$P_B \rightarrow C \quad C \neq B$$

resultado indicando que se ha obtenido el resultado deseado.

$$C \leftarrow 0$$

## Práctica 3:

### Práctica 3

Recuperar el mensaje original a partir del código interpretado.

Pasos para decodificar un mensaje:

Para decodificar un mensaje, primero debemos observar una de las "reglas" que se nos dan:

La interpretación del símbolo inicial es  $\begin{cases} 0 \rightarrow 0 \\ 1 \rightarrow 1 \end{cases}$ , es decir,

si como primer símbolo nos entra un 0, su interpretación es 0. Si como primer símbolo nos entra un 1, su interpretación es 1.

Para el resto de la cadena entrante, la interpretación es la siguiente:

Debemos fijarnos en el valor escrito anteriormente. Si el valor que hemos escrito anteriormente es un 0, sus interpretaciones serían: Si nos entra un 0, el valor de salida sería 0. Si nos entra un 1, el valor de salida sería 1.

Para el caso en el que el valor escrito anterior es 1, su interpretación es la siguiente:

Si el valor de entrada es un 0, el valor de salida sería 1. Si el valor de entrada es 1, el valor de salida sería 0.

Para la interpretación del valor de salida, debemos fijarnos principalmente en el valor escrito anterior, que es el que determinará el cambio de estado.

Para ello vamos a interpretar un mensaje codificado y vamos a usar las reglas anteriores para decodificarlo:

Entrada: 1 0 1 1 0 1 0 0 1

- El primer símbolo es un 1, así es que la salida será 1.
- El siguiente es un 0, y nuestro valor escrito anterior es 1, según la regla, el valor de salida sería 1.
- El siguiente es un 1, y el anterior es 1, por tanto según la regla el valor de salida sería 0.
- El siguiente es un 1 y el anterior escrito es 0. El valor de salida sería 1.
- El siguiente es un 0 y el anterior escrito es 1. El valor de salida es 1.
- El siguiente es un 1 y el anterior escrito es 1. El valor de salida es 0.
- El siguiente es un 0 y el anterior escrito es 0. El valor de salida es 0.
- El siguiente es un 0 y el anterior escrito es 0. El valor de salida es 0.
- El siguiente es un 1 y el anterior escrito es 0. El valor de salida es 1.

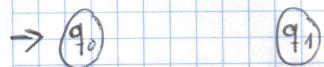
Por tanto el mensaje de salida decodificado sería:

Salida: 1 1 0 1 1 0 0 0 1

Para representar el autómata, necesitaremos dos estados:

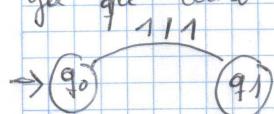
$q_0$ : Representa a valor escrito anterior 0 y será nuestro estado inicial.

$q_1$ : Representa a valor escrito anterior 1.

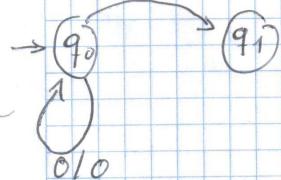


Nos fijamos en las reglas para determinar los cambios de estado.

Si partimos de  $q_0$ , cambiamos de estado cuando entra un 1 ya que cuando anterior escrito es 0, la salida es 1.



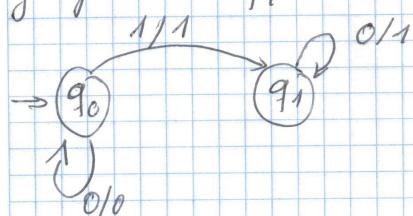
Cuando en  $q_0$  nos entra un 0, seguimos en  $q_0$  ya que cuando entra 0 y ant. escrito es 0, la salida es 0.



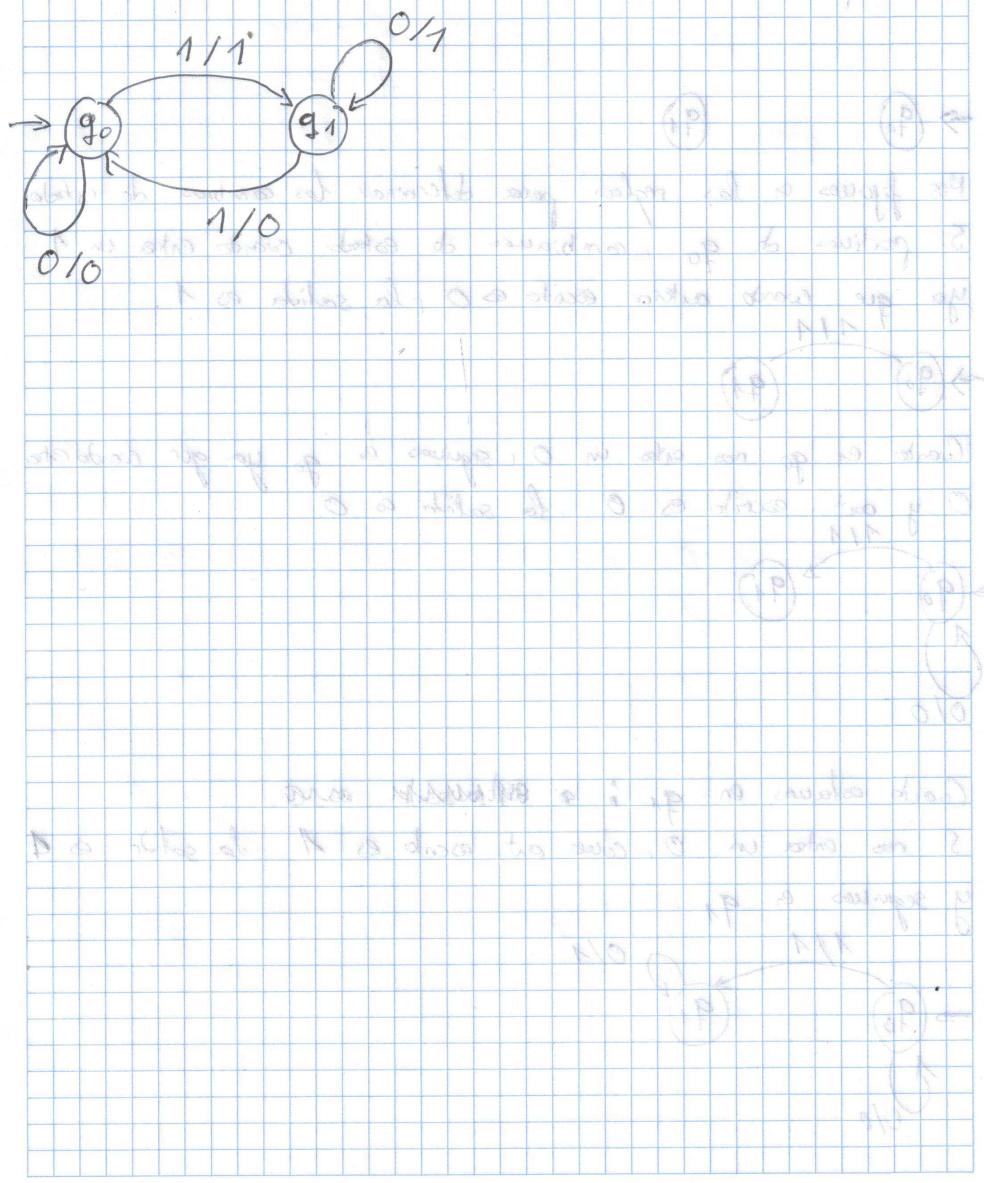
Cuando estamos en  $q_1$ :

Si nos entra un 0, como ant. escrito es 1 la salida es 0

y seguimos en  $q_1$ .



Si nos entra un 1, como así escribo el 1, se producirá cambio de estado y pasaremos a  $q_1$ .  
 Por tanto ya tenemos nuestro automata completo para representar nuestro decodificador.



## Práctica 4:

# PROYECTO LEX PRACTICAS MC

---

### 1º Descripción del proyecto:

Queremos realizar un listado de una lista de productos (nombre y precio). Para ello usaremos un listado de productos de la web pccomponentes (ordenadores) y hacer uso de Lex y expresiones regulares.

### 2º Resolución del problema:

Realizamos una búsqueda en el html de la web y localizamos las etiquetas “data-name” (contienen el nombre del producto) y “data-price”(contienen el precio del producto).

Cuando encontramos “data-name”, realizamos un salto para detectar la cadena que contiene el nombre entre comillas.

Después localizamos “data-price” y realizamos otro salto para detectar el número (real) que será el precio del artículo.

Una vez realizado esto, debemos volver al principio, para no detectar ningún otro número y mostrarlo por pantalla.

### 3º Parte del código:

```
/*---- Sección de Declaraciones -----*/  
%{ #include<stdio.h> %}  
  
cadena1 "data-name="  
  
cadena2 "data-price="  
  
identificacion ((\").(.)+("))  
  
digito [0-9]  
  
suc ({digito}+)  
  
entero ({suc})  
  
real ({entero}{\,}?({digito})*)  
  
/*---- Sección de Reglas -----*/
```

```
%START AA BB

%%

{cadena1} {

    printf("Id_Articulo: ");BEGIN AA;

}

{cadena2} {

    printf("Precio: ");BEGIN BB;

}

<AA>{identificacion} {printf(" %s ; ",yytext);}

<BB>{real} {printf(" %s \n",yytext);BEGIN 0;}

.|\\n {}

%%

yywrap(){

    printf("Listado completado\n");

    return 1;

}
```

La salida de la ejecución de mi proyecto de lex es el siguiente:

```
Id_Articulo: "MSI WS60 20J-279XES i7-4720HQ/16GB/1TB+128SSD/K2100M/15.6" ; Precio: 1999 €
Id_Articulo: "MSI WS60 20J-279XES i7-4720HQ/16GB/1TB+128SSD/K2100M/15.6" ; Precio: 1999 €
Id_Articulo: "Apple MacBook Pro Retina Display Intel i7/16GB/256GB/15.4" ; Precio: 2059 €
Id_Articulo: "MSI GS70 6QE-024ES i7-6700/16GB/1TB+256 SSD/GTX970M/17.3" ; Precio: 2099 €
Id_Articulo: "MSI GS70 6QE-024ES i7-6700/16GB/1TB+256 SSD/GTX970M/17.3" ; Precio: 2099 €
Id_Articulo: "MSI GT72 6QD-091ES i7-6700HQ/16GB/1TB+128SSD/GTX970M/17.3" ; Precio: 2099 €
Id_Articulo: "MSI GT72 6QD-091ES i7-6700HQ/16GB/1TB+128SSD/GTX970M/17.3" ; Precio: 2099 €
Id_Articulo: "MSI WT72 20M-1278XES i7-4720HQ/16GB/1TB/K2200M/17.3" ; Precio: 2099 €
Id_Articulo: "MSI WT72 20M-1278XES i7-4720HQ/16GB/1TB/K2200M/17.3" ; Precio: 2099 €
Id_Articulo: "ASUS G750JY-T4033H i7-4710HQ/16GB/1TB+256GB/GTX980M/17.3" ; Precio: 2119 €
Id_Articulo: "Asus G750JY-T4033H i7-4710HQ/16GB/1TB+256GB/GTX980M/17.3" ; Precio: 2119 €
Id_Articulo: "Asus G750JY-T4051H i7-4720HQ/32GB/2TB+512SSD/GTX980M/17.3" ; Precio: 2189 €
Id_Articulo: "Asus G750JY-T4051H i7-4720HQ/32GB/2TB+512SSD/GTX980M/17.3" ; Precio: 2189 €
Id_Articulo: "Gigabyte P37X V3 i7-4720HQ/16GB/1TB+128GB SSD/GTX980M/17.3" ; Precio: 2199 €
Id_Articulo: "Gigabyte P37X V3 i7-4720HQ/16GB/1TB+128GB SSD/GTX980M/17.3" ; Precio: 2199 €
Id_Articulo: "MSI GT80 2QE-473XES i7-5700HQ/16GB/1TB+256SSD/GTX980M/18.4" ; Precio: 2299 €
Id_Articulo: "MSI GT80 2QE-473XES i7-5700HQ/16GB/1TB+256SSD/GTX980M/18.4" ; Precio: 2299 €
Id_Articulo: "MSI GS30 2M-033ES i7-4780HQ/16GB/512GB SSD/GTX970/13.3" ; Precio: 2399 €
Id_Articulo: "MSI GS30 2M-033ES i7-4780HQ/16GB/512GB SSD/GTX970/13.3" ; Precio: 2399 €
Id_Articulo: "MSI GT72S 6QD-085ES i7-6820HK/16GB/1TB+256SSD/GTX970M/17.3" ; Precio: 2399 €
Id_Articulo: "MSI GT72S 6QD-085ES i7-6820HK/16GB/1TB+256SSD/GTX970M/17.3" ; Precio: 2399 €
Id_Articulo: "MSI WT72 20M-1274ES i7-4720HQ/16GB/1TB+128SSD/K2200M/17.3" ; Precio: 2399 €
Id_Articulo: "MSI WT72 20M-1274ES i7-4720HQ/16GB/1TB+128SSD/K2200M/17.3" ; Precio: 2399 €
Id_Articulo: "Apple MacBook Pro Retina Display i7/16GB/512GB/15" Reacondicionado" ; Precio: 2482 €
Id_Articulo: "Apple MacBook Pro Retina Display i7/16GB/512GB/15" Reacondicionado" ; Precio: 2482 €
Id_Articulo: "MSI WS60 20J-087ES i7-4720HQ/16GB/1TB+128SSD/K2100M/15.6" 4K" ; Precio: 2489 €
Id_Articulo: "MSI WS60 20J-087ES i7-4720HQ/16GB/1TB+128SSD/K2100M/15.6" 4K" ; Precio: 2489 €
Id_Articulo: "MSI GT72 2QE-1622ES i7-5700HQ/16GB/1TB+512 SSD/GTX980M/17.3" ; Precio: 2499 €
Id_Articulo: "MSI GT72 2QE-1622ES i7-5700HQ/16GB/1TB+512 SSD/GTX980M/17.3" ; Precio: 2499 €
Id_Articulo: "MSI GT80 2QD-472XES i7-5700HQ/16GB/1TB+128GB/2xGTX970M/18.4" ; Precio: 2499 €
Id_Articulo: "MSI GT80 2QD-472XES i7-5700HQ/16GB/1TB+128GB/2xGTX970M/18.4" ; Precio: 2499 €
Id_Articulo: "MSI WT72 20K-1273XES i7-4720HQ/16GB/1TB/K3100M/17.3" ; Precio: 2499 €
Id_Articulo: "MSI WT72 20K-1273XES i7-4720HQ/16GB/1TB/K3100M/17.3" ; Precio: 2499 €
Id_Articulo: "Apple MacBook Pro Retina Display i7/16GB/512GB/15" ; Precio: 2559 €
Id_Articulo: "Apple MacBook Pro Retina Display i7/16GB/512GB/15" ; Precio: 2559 €
Id_Articulo: "MSI GT72S 6QE-079ES i7-6820HK/16GB/1TB+256SSD/GTX980M/17.3" ; Precio: 2699 €
Id_Articulo: "MSI GT72S 6QE-079ES i7-6820HK/16GB/1TB+256SSD/GTX980M/17.3" ; Precio: 2699 €
Id_Articulo: "MSI WT72 20K-1270ES i7-4720HQ/16GB/1TB+128SSD/K3100M/17.3" ; Precio: 2799 €
Id_Articulo: "MSI WT72 20K-1270ES i7-4720HQ/16GB/1TB+128SSD/K3100M/17.3" ; Precio: 2799 €
Id_Articulo: "GT72S 6QE-484ES Heroes Ed. i7-6820HK/16GB/1TB+256SSD/GTX980M/17.3" ; Precio: 2849 €
Id_Articulo: "GT72S 6QE-484ES Heroes Ed. i7-6820HK/16GB/1TB+256SSD/GTX980M/17.3" ; Precio: 2849 €
Id_Articulo: "MSI GT72 2QE-293ES i7-4980HQ/32GB/1TB+512 SSD/GTX980M/17.3" ; Precio: 2899 €
Id_Articulo: "MSI GT72 2QE-293ES i7-4980HQ/32GB/1TB+512 SSD/GTX980M/17.3" ; Precio: 2899 €
Id_Articulo: "MSI GT72 2QE-697ES i7-4980HQ/32GB/1TB+512SSD/GTX980M/17.3" ; Precio: 2999 €
Id_Articulo: "MSI GT72 2QE-697ES i7-4980HQ/32GB/1TB+512SSD/GTX980M/17.3" ; Precio: 2999 €
Id_Articulo: "MSI WT72 20K-1266ES i7-4720HQ/32GB/1TB+128SSD/K4100M/17.3" ; Precio: 2999 €
Id_Articulo: "MSI WT72 20K-1266ES i7-4720HQ/32GB/1TB+128SSD/K4100M/17.3" ; Precio: 2999 €
Id_Articulo: "MSI GT80 2QE-239XES i7-4720HQ/16GB/1TB+128GB/2xGTX980M/18.4" ; Precio: 3099 €
Id_Articulo: "MSI GT80 2QE-239XES i7-4720HQ/16GB/1TB+128GB/2xGTX980M/18.4" ; Precio: 3099 €
Id_Articulo: "MSI GT72 2QE-1263ES i7-4980HQ/32GB/1TB+512SSD/GTX980M/17.3" ; Precio: 3159 €
Id_Articulo: "MSI GT72 2QE-1263ES i7-4980HQ/32GB/1TB+512SSD/GTX980M/17.3" ; Precio: 3159 €
Id_Articulo: "MSI GT72S 6QF-037ES i7-6820HK/32GB/1TB+512SSD/GTX980/17.3" ; Precio: 3699 €
Id_Articulo: "MSI GT72S 6QF-037ES i7-6820HK/32GB/1TB+512SSD/GTX980/17.3" ; Precio: 3699 €
Id_Articulo: "MSI GT80 2QE-044ES i7-4980HQ/32GB/1TB+512SSD/2xGTX980M/18.4" ; Precio: 4299 €
Id_Articulo: "MSI GT80 2QE-044ES i7-4980HQ/32GB/1TB+512SSD/2xGTX980M/18.4" ; Precio: 4299 €
```

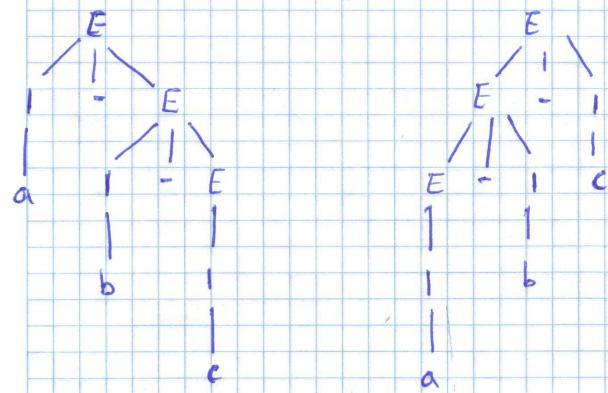
Listado completado

## Práctica 5:

### Práctica 5

Dado la gramática  $E \rightarrow I$   $E \rightarrow I-E$   $E \rightarrow E-I$   $I \rightarrow a b c l d$   
¿Hay una gramática que genere la misma cadena no ambigua?

Como puedes observar, la gramática dada, es ambigua ya que admite dos árboles de derivación distintos de a-b-c.



Para encontrar una gramática, que genere la misma cadena, no ambigua, no fijarnos en las regulares producciones. Como puedes observar, si eliminamos la producción  $E \rightarrow I-E$ , la gramática dejó de ser ambigua.