

MODELOS DE COMPUTACIÓN (2016-2017)  
GRADO EN INGENIERÍA INFORMÁTICA  
UNIVERSIDAD DE GRANADA

---

## Memoria de practicas

---



Antonio Rodríguez Alaminos

29 de enero de 2017

# Índice

<b>1</b>	<b>Práctica 1</b>	<b>4</b>
1.1	Ejercicio 1 . . . . .	4
1.1.1	Apartado A . . . . .	4
1.1.2	Apartado B . . . . .	4
1.1.3	Apartado C . . . . .	5
1.2	Ejercicio 2 . . . . .	5
1.2.1	Apartado A . . . . .	5
1.2.2	Apartado B . . . . .	5
1.2.3	Apartado C . . . . .	5
1.3	Ejercicio 3 . . . . .	6
<b>2</b>	<b>Practica 2</b>	<b>7</b>
2.1	Ejercicio 1 . . . . .	7
2.1.1	Apartado A . . . . .	7
2.1.2	Apartado B . . . . .	7
2.1.3	Apartado C . . . . .	8
2.2	Ejercicio 2 . . . . .	8
2.2.1	Apartado A . . . . .	8
2.2.2	Apartado B . . . . .	9
2.2.3	Apartado C . . . . .	9
2.3	Ejercicio 3 . . . . .	10
2.4	Ejercicio 4 . . . . .	11
<b>3</b>	<b>Practica 3</b>	<b>12</b>
3.1	Ejercicio 1 . . . . .	12
3.2	Ejercicio 2 . . . . .	12
3.3	Ejercicio 3 . . . . .	16
<b>4</b>	<b>Practica 4</b>	<b>18</b>
4.1	Ejercicio 1 . . . . .	18
4.1.1	Apartado A . . . . .	18
4.1.2	Apartado B . . . . .	18
4.1.3	Apartado C . . . . .	19
4.2	Ejercicio 2 . . . . .	19
4.3	Ejercicio 3 . . . . .	23
4.4	Ejercicio 4 . . . . .	25

## Índice de figuras

2.1.	prac 2 - ejer 1A . . . . .	7
2.2.	prac 2 - ejer 1B . . . . .	7
2.3.	prac 2 - ejer 1C . . . . .	8

2.4. prac 2 - ejer 2A . . . . .	8
2.5. prac 2 - ejer 2B . . . . .	9
2.6. prac 2 - ejer 2C . . . . .	9
2.7. prac 2 - ejer 2C . . . . .	10
2.8. prac 2 - ejer 2C . . . . .	11

## Índice de tablas

2.1. prac 2 - ejer 2C . . . . .	11
---------------------------------	----

## 1. Práctica 1

### 1.1. Ejercicio 1

Describir el lenguaje generado por las siguientes gramáticas en  $\{0, 1\}^*$ .

#### 1.1.1. Apartado A

$$S \rightarrow 0S_11 \qquad S_1 \rightarrow 0S_1|1S_1|\epsilon$$

Cuando tenemos que  $S_1 \rightarrow \epsilon$  :

$$S \rightarrow 0S_11 \Rightarrow 0\epsilon1 \Rightarrow \mathbf{01}$$

Ahora comprobamos cuando  $S_1 \rightarrow 0S_1$  :

$$S \rightarrow 0S_11 \rightarrow 00S_11 \rightarrow 000S_11 \rightarrow \mathbf{00000...1}$$

Continuamos cuando  $S_1 \rightarrow 1S_1$  :

$$S \rightarrow 0S_11 \Rightarrow 01S_11 \Rightarrow 01S_111 \Rightarrow \mathbf{0...11111}$$

Y por ultimo cuando tenemos que  $S_1 \rightarrow 1S_1$  o  $S_1 \rightarrow 0S_1$ :

$$S \rightarrow 0S_11 \Rightarrow 01S_11 \Rightarrow 01S_111 \Rightarrow 0S_1111 \Rightarrow 000S_1111 \Rightarrow \mathbf{0...0101...1010...1}$$

Por lo tanto tenemos que  $L_n = \{0u1 \text{ tq } u \in \{0,1\}^*\}$

#### 1.1.2. Apartado B

$$S \rightarrow S_1101S_1 \qquad S_1 \rightarrow 0S_1|1S_1|\epsilon$$

Aplicamos los mismos pasos del ejercicio anterior, empezamos con  $S_1 \rightarrow \epsilon$  :

$$S \rightarrow S_1101S_1 \Rightarrow \epsilon101\epsilon \Rightarrow \mathbf{101}$$

Ahora comprobamos cuando  $S_1 \rightarrow 0S_1$  :

$$S \rightarrow S_1101S_1 \Rightarrow 0S_11010S_1 \Rightarrow 0S_1010100S_1 \Rightarrow \mathbf{00000...101...00000}$$

Continuamos cuando  $S_1 \rightarrow 1S_1$  :

$$S \rightarrow S_1101S_1 \Rightarrow 1S_11011S_1 \Rightarrow 1S_1110111S_1 \Rightarrow \mathbf{11111...101...11111}$$

Y por ultimo cuando tenemos que  $S_1 \rightarrow 1S_1$  o  $S_1 \rightarrow 0S_1$ :

$$S \rightarrow S_1101S_1 \Rightarrow 1S_11011S_1 \Rightarrow 1S_1110111S_1 \Rightarrow 0S_111101110S_1 \Rightarrow 0S_10111011100S_1 \Rightarrow \mathbf{0101....101...1010}$$

Por lo tanto tenemos que  $L_n = \{u101v \text{ tq } u, v \in \{0,1\}^*\}$

### 1.1.3. Apartado C

$$S \rightarrow 0S_1|S_1 \quad || \quad S_1 \rightarrow 1S_10|1S_20 \quad || \quad S_2 \rightarrow 0S_21|\epsilon$$

Por lo tanto tenemos que  $L_n = \{0^i 1^j 0^k 1^k 0^j \text{ tq } i \in [0, 1], j \geq 1, k \geq 0\}$

## 1.2. Ejercicio 2

Encontrar gramáticas de tipo 2 para los siguientes lenguajes sobre el alfabeto  $\{0,1\}$ . En cada caso determinar si los lenguajes generados son de tipo 3, estudiando si existe una gramática de tipo 3 que los genera.

### 1.2.1. Apartado A

Palabras que comienzan con la subcadena "10" y acaban en "001".

#### ■ Tipo 2

$$G = \{S, \{A\}, \{01\}, P, S\}$$

$$P = \{S \rightarrow 10A001 \quad A \rightarrow 1A \quad A \rightarrow 0A \quad A \rightarrow \epsilon\}$$

#### ■ Tipo 3

$$G = \{S, \{S_1, S_2\}, \{01\}, P, S\}$$

$$P = \{S \rightarrow 10S_1|S_2 \quad S_1 \rightarrow 1S_1|0S_1|S_2 \quad S_2 \rightarrow 001\}$$

### 1.2.2. Apartado B

Palabras que tienen 2 o 3 "0".

#### ■ Tipo 2

$$G = \{S, \{A, B, C\}, \{01\}, P, S\}$$

$$P = \{S \rightarrow 1S|0S_1|0S_30 \quad S_1 \rightarrow 1S_1|0S_2|0S_3 \quad S_2 \rightarrow 1S_2|0S_3 \quad S_3 \rightarrow \epsilon\}$$

#### ■ Tipo 3

$$G = \{S, \{A, B, C\}, \{01\}, P, S\}$$

$$P = \{S \rightarrow 1A \quad S \rightarrow 0B \quad A \rightarrow 1A \quad A \rightarrow 0B \quad B \rightarrow 1B \\ B \rightarrow 0C \quad C \rightarrow 1C \quad C \rightarrow 0D \quad C \rightarrow \epsilon \quad D \rightarrow 1D \quad D \rightarrow \epsilon\}$$

### 1.2.3. Apartado C

Palabras que no contienen la subcadena "011".

#### ■ Tipo 3

$$G = \{S, \{A, B\}, \{01\}, P, S\}$$

$$P = \{S \rightarrow 1S \quad S \rightarrow 0A \quad S \rightarrow \epsilon \quad A \rightarrow 0A \quad A \rightarrow 1B \\ A \rightarrow \epsilon \quad B \rightarrow 0S \quad B \rightarrow \epsilon\}$$

### 1.3. Ejercicio 3

Como empleado de la empresa de desarrollo de videojuegos “MoreThanDungeons”, se le ha pedido diseñar una gramática que represente los niveles de un juego de exploración de mazmorras y las salas de estas, con una serie de restricciones.

En cada nivel:

- Existen salas grandes ( $g$ ) y pequeñas ( $p$ ) que deberán ser limpiadas de monstruos para avanzar. (Los niveles más sencillos tienen al menos una sala grande)
- Hay al menos una sala de tendero ( $t$ ), donde recuperar fuerzas y comprar objetos.
- Habrá una sola sala secreta ( $x$ ), siempre le precede una sala grande. Es decir, siempre habrá una “ $g$ ” delante de “ $x$ ”.
- Cada nivel de la mazmorra debe acabar con una sala final de jefe ( $j$ ).

Por ejemplo, la cadena terminal “ppgxtj” representa el nivel en el que el jugador debe de pasar por dos habitaciones pequeñas “pp”, seguidas de una grande “g”. En esta, podrá encontrar la sala secreta “x”. A continuación, podrá recuperar fuerzas en la tienda “t”. Para finalmente, enfrentarse al jefe final “j” del nivel.

Elabore una gramática que genere estos niveles con sus restricciones. Cada palabra del lenguaje es UN SOLO NIVEL. ¿A qué tipo de la jerarquía de Chomsky pertenece la gramática que ha diseñado?

$$G = \{S, A, \{g, x, t, p\}, P, S\}$$

$$P = \{S \rightarrow AtAgxAj \quad A \rightarrow gA \quad A \rightarrow pA \quad A \rightarrow tA \quad A \rightarrow \epsilon\}$$

¿Podría diseñar una gramática de tipo 3 para dicho problema?

$$G = \{S, A, \{g, x, t, p\}, P, S\}$$

$$P = \{S \rightarrow gS|pS|tS_1|gS_2 \quad S_1 \rightarrow gS_1|pS_1|tS_1|gS_3 \quad S_2 \rightarrow gS_2|pS_2|tS_3$$

$$S_3 \rightarrow gS_3|pS_3|tS_3|j\}$$

## 2. Practica 2

### 2.1. Ejercicio 1

Construir un AFD que acepte cada uno de los siguientes lenguajes con alfabeto  $\{a,b\}$ :

#### 2.1.1. Apartado A

El lenguaje de las palabras que contienen la subcadena aaa.

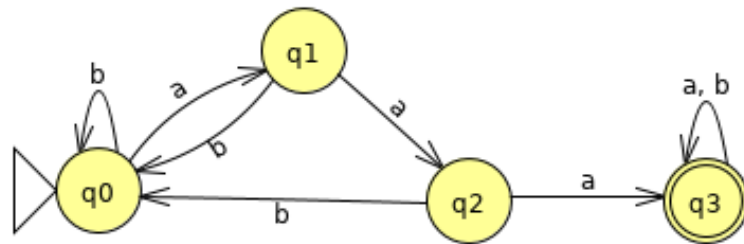


Figura 2.1: Grafo del ejercicio 1A

#### 2.1.2. Apartado B

El lenguaje de las palabras que empiezan o terminan (o ambas cosas) en aaa.

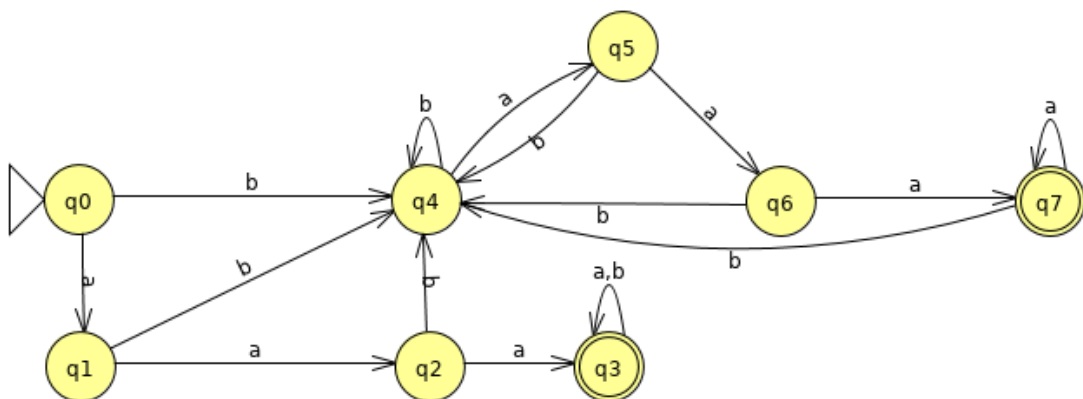


Figura 2.2: Grafo del ejercicio 1B

### 2.1.3. Apartado C

El lenguaje formado por las cadenas donde el número de  $a$ 's es divisible por 3.

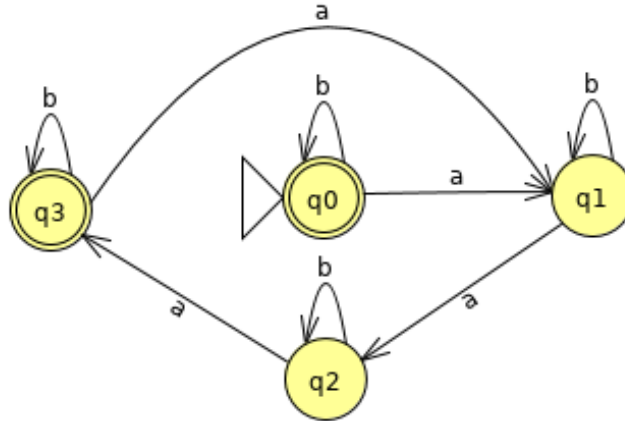


Figura 2.3: Grafo del ejercicio 1C

## 2.2. Ejercicio 2

Construir un AFND que acepte cada uno de los siguientes lenguajes con alfabeto  $\{a,b\}$ :

### 2.2.1. Apartado A

El lenguaje de las palabras que terminan en  $aaa$ .

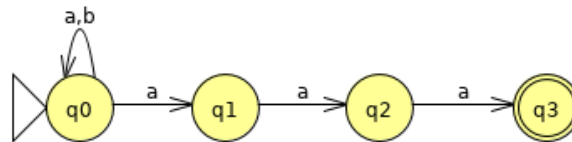


Figura 2.4: Grafo del ejercicio 2A



### 2.2.2. Apartado B

El lenguaje de las palabras que empiezan o terminan (o ambas cosas) en  $aaa$ .

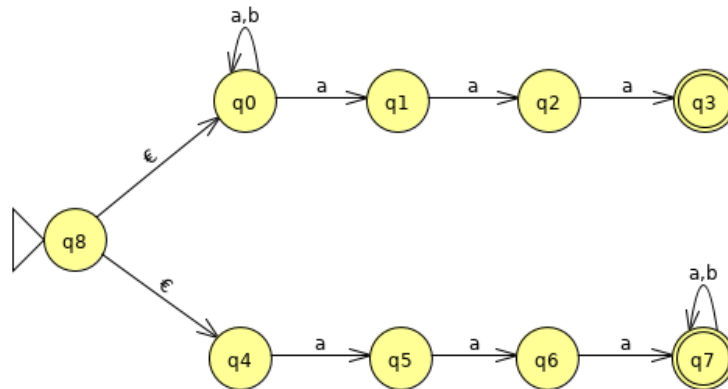


Figura 2.5: Grafo del ejercicio 2B

### 2.2.3. Apartado C

El lenguaje de las palabras que contengan, simultáneamente, las subcadenas  $aba$  y  $abb$ . Este AFND también acepta cadenas en la que estas subcadenas están solapadas (por ejemplo, las palabras “ $ababb$ ” y “ $aaabbbaba$ ” serían aceptadas).

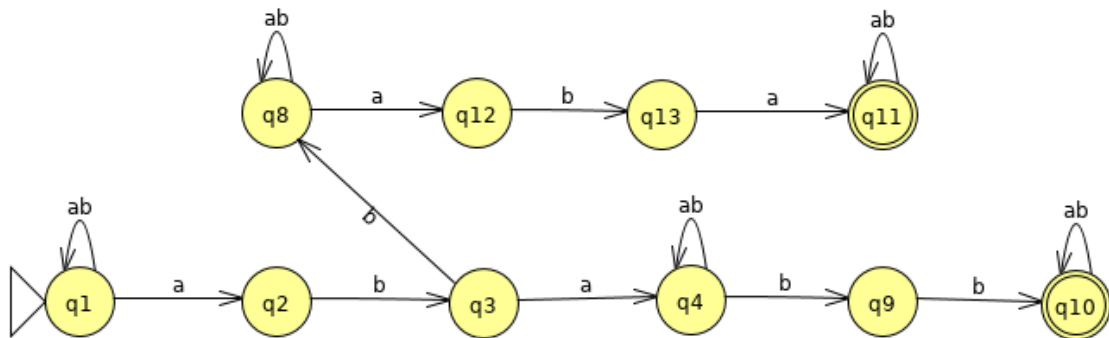


Figura 2.6: Grafo del ejercicio 2C

### 2.3. Ejercicio 3

Diseñar una Máquina de Mealy o de Moore que, dada una cadena usando el alfabeto  $A = \{a, w, o\}$ , encienda un led verde (salida 'V') cada vez que se detecte la cadena "woow" en la entrada, apagándolo cuando lea cualquier otro símbolo después de esta cadena (representamos el led apagado con la salida "X"). El autómata tiene que encender el led verde (salida 'V'), tantas veces como aparezca en la secuencia "woow" en la entrada, y esta secuencia puede estar solapada. Por ejemplo, ante la siguiente entrada, la Máquina de Mealy/Moore emitirá la salida:

<b>Entrada</b>	aaawoawoowwoowwoowa
<b>Salida</b>	XXXXXXXXXXVXXVXXXVX

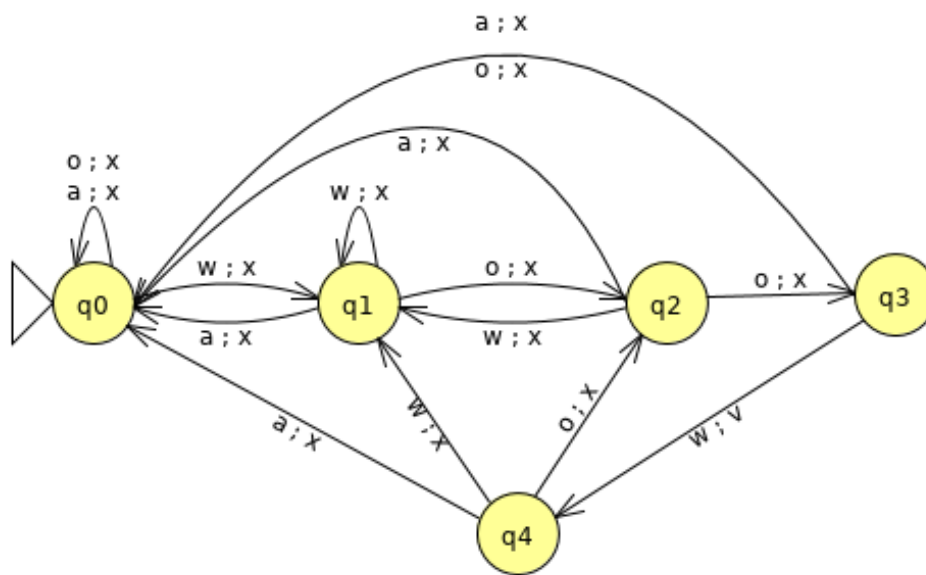
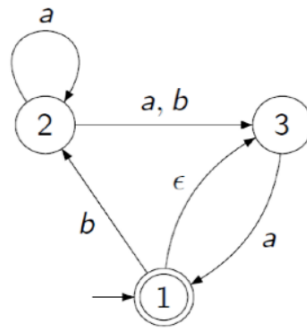


Figura 2.7: Grafo del ejercicio 3

## 2.4. Ejercicio 4

Obtener un AFD equivalente al AFND siguiente:



Estado/Atributo	a	b
$q_1q_3$	$q_1q_3$	$q_2$
$q_2$	$q_2q_3$	$q_3$
$q_2q_3$	$q_2q_3q_1$	$q_3$
$q_2q_3q_1$	$q_2q_3q_1$	$q_3q_2$
$q_3$	$q_1q_3$	$\emptyset$

Tabla 2.1: Caminos, ejercicio 4

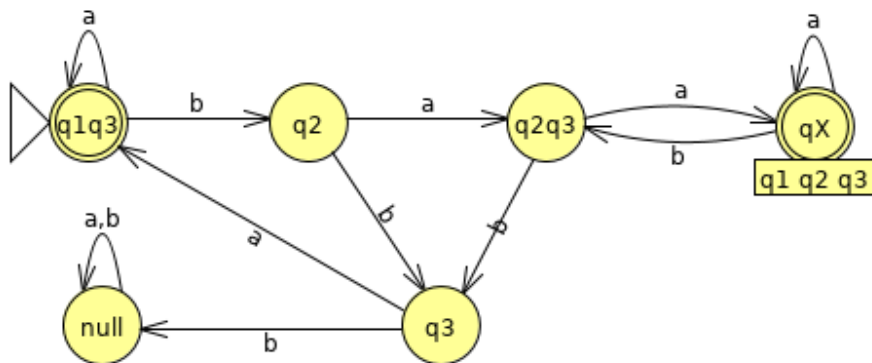


Figura 2.8: Grafo del ejercicio 4

### 3. Practica 3

#### 3.1. Ejercicio 1

*Pensar un problema original de procesamiento de textos. Para la resolución de este problema debe ser apropiado el uso de Lex, o sea, se debe resolver mediante el emparejamiento de cadenas con expresiones regulares y la asociación de acciones a cada emparejamiento. Se presentará una descripción por escrito del problema. Consultar al profesor de prácticas acerca de la complejidad del problema propuesto.*

El problema real que voy a resolver, trata en obtener: las cadenas que tienen la información relevante sobre los log obtenidos tras una investigación de clasificación sobre datos no balanceados.

#### 3.2. Ejercicio 2

*Resolver el problema propuesto usando Lex.*

Código del autómata en lex.

```
/*----- Seccion de Declaraciones -----*/

%{

#include <stdio.h>
#include <string.h>
#include <stdbool.h>

int tiempoIntervalo = 0;
int tiempo = 0;
int contador = 1;
char charTimeInit[11];
char charTimeEnd[11];
void escribirNombre(char* dato1, int tam);
void escribirInicio(char* dato1, int tam);
void escribirFin(char* dato1, int tam);

}%

initTime      ( "***_BEGIN_OF_EXPERIMENT_" )
fechaDia      { initTime } ([a-zA-Z]{3})
fechaMes      { fechaDia } . ([a-zA-Z]{3})
fechaDiaNum   { fechaMes } . ([0-9]{2})
hora          { fechaDiaNum } . ([0-9]{2}:[0-9]{2}:[0-9]{2})
cet           { hora } . ([a-zA-Z]{3})
initProceso   { cet } . ([0-9]{4})

initNombre    ( "The_name_is:_" )
nombreProceso { initNombre } ([a-zA-Z]+) . ([a-zA-Z]+)*
```

```

endTime          ( "***_END_OF_EXPERIMENT_" )
endfechaDia      { endTime } ( [a-zA-Z]{3} )
endfechaMes      { endfechaDia } ( [a-zA-Z]{3} )
endfechaDiaNum   { endfechaMes } ( [0-9]{2} )
endhora          { endfechaDiaNum } ( [0-9]{2} \:[0-9]{2} \:[0-9]{2} )
endcet           { endhora } ( [a-zA-Z]{3} )
endProceso       { endcet } ( [0-9]{4} )

%%

/*----- Seccion de Reglas -----*/

{initProceso}      { escribirInicio(yytext, yyleng); }
{nombreProceso}    { escribirNombre(yytext, yyleng); }
{endProceso}       { escribirFin(yytext, yyleng); }
.                  {}
\n                 {}

%%

/*----- Seccion de Procedimientos -----*/

int main (int argc, char *argv[]) {
    if (argc == 2) {
        yyin = fopen (argv[1], "rt");

        if (yyin == NULL) {
            printf ("El_fichero_%s_no_se_puede_abrir\n",
                    argv[1]);
            exit (-1);
        }
    }
    else yyin = stdin;

    yylex ();
    printf ("\n");
    printf ("\n-->_Tiempo_Final:_%s", tiempo);
    printf ("\n");

    return 0;
}

void escribirInicio(char* datol, int tam){
    char* tipo = "Inicio:";
    char* aux;
    char letra;
    int tamano = 32;

    printf ("\n##_Inicio_Proceso:_%a##_\n", contador);
    printf ("\n%s", tipo);

    aux = datol;

```

```

        for(int i=24; i < strlen(dato1); i++){
            letra = dato1[i];
            printf ("%c",letra);
        }

        for(int i=0; i < 11; i++){
            letra = aux[tamano];
            charTimeInit[i] = letra;
            tamano++;
        }
    }

void escribirFin(char* dato1, int tam){
    char* tipo = "Fin: ";
    char letra;
    char* aux;
    int tamano = 30;

    if (tieneNombre != true){
        printf ("\nNombre: No tiene nombre.");
    }

    printf ("\n%s",tipo);

    aux = dato1;

    for(int i=22; i < strlen(dato1); i++){
        letra = dato1[i];
        printf ("%c",letra);
    }

    for(int i=0; i < 11; i++){
        letra = aux[tamano];
        charTimeEnd[i] = letra;
        tamano++;
    }

    tiempoTotal();

    printf ("\n\n## Fin_Proceso: %d ##\n\n",
            #####"\n", contador);
    \\

    tieneNombre = false;

    contador++;
}

void escribirNombre(char* dato1, int tam){
    char* tipo = "Nombre: ";
    char letra;

    printf ("\n%s",tipo);

```

```

        for(int i=13; i < strlen(dato1); i++){
            letra = dato1[i];
            printf ("%c",letra);
        }

        tieneNombre = true;
    }

    void tiempoTotal(){
        int tamano = 0, tiempoTotalInit = 0, tiempoTotalEnd = 0;
        char CdiaInit[2], CdiaEnd[2], ChoraInit[2], ChoraEnd[2],
            CminInit[2], CminEnd[2], CsegInit[2], CsegEnd[2];

        for(int i = 0; i < 2; i++){
            CdiaInit[i] = charTimeInit[tamano];
            CdiaEnd[i] = charTimeEnd[tamano];
            tamano++;
        }

        tamano++;

        for(int i = 0; i < 2; i++){
            ChoraInit[i] = charTimeInit[tamano];
            ChoraEnd[i] = charTimeEnd[tamano];
            tamano++;
        }

        tamano++;

        for(int i = 0; i < 2; i++){
            CminInit[i] = charTimeInit[tamano];
            CminEnd[i] = charTimeEnd[tamano];
            tamano++;
        }

        tamano++;

        for(int i = 0; i < 2; i++){
            CsegInit[i] = charTimeInit[tamano];
            CsegEnd[i] = charTimeEnd[tamano];
            tamano++;
        }

        tiempoTotalInit = atoi(CdiaInit)*24*60*60;
        tiempoTotalInit += atoi(ChoraInit)*60*60;
        tiempoTotalInit += atoi(CminInit)*60;
        tiempoTotalInit += atoi(CsegInit);

        tiempoTotalEnd = atoi(CdiaEnd)*24*60*60;
        tiempoTotalEnd += atoi(ChoraEnd)*60*60;
        tiempoTotalEnd += atoi(CminEnd)*60;
        tiempoTotalEnd += atoi(CsegEnd);

        tiempoIntervalo = tiempoTotalEnd - tiempoTotalInit;
    }

```

```

        tiempo += tiempoIntervalo;

        printf ( "\n-->Tiempo_Proceso:_%s", tiempoIntervalo);
    }

```

Enlaces de descarga:

- [Automata lex.](#)
- [Prueba 1](#)
- [Prueba 2](#)
- [Resultado](#)

### 3.3. Ejercicio 3

*Realizar un documento presentando el problema y la solución con Lex.*

El principal problema que tiene el extraer esta información sin ayuda de los autómatas es la cantidad de tiempo y lo compleja que encontrarla entre mas de 45000 de líneas de código.

Solución obtenida con el autómatas en diversos casos:

- **Prueba 20**

```
## Inicio Proceso: 1 ##
```

```

Inicio: Fri Jan 06 00:51:34 CET 2017
Nombre: IterativePartitioningFilter
Fin: Fri Jan 06 01:14:07 CET 2017
->Tiempo Proceso: 1353s

```

```
## Fin Proceso: 1 ##
```

```
#####
```

```
## Inicio Proceso: 2 ##
```



Inicio: Fri Jan 06 01:14:07 CET 2017  
Nombre: IterativePartitioningFilter  
Fin: Fri Jan 06 01:36:40 CET 2017  
->Tiempo Proceso: 1353s

## Fin Proceso: 2 ##

#####

.....

#####

## Inicio Proceso: 31 ##

Inicio: Fri Jan 06 03:29:08 CET 2017  
Nombre: No tiene nombre.  
Fin: Fri Jan 06 03:29:08 CET 2017  
->Tiempo Proceso: 0s

## Fin Proceso: 31 ##

#####

->Tiempo Final: 9454s

## 4. Practica 4

### 4.1. Ejercicio 1

*Determinar cuáles de las siguientes gramáticas son ambiguas y, en su caso, comprobar si los lenguajes generados son inherentemente ambiguos, Justificar la respuesta.*

#### 4.1.1. Apartado A

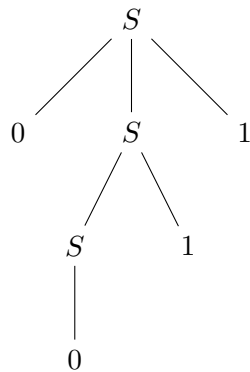
$$S \rightarrow 01S \mid 010S \mid 101S \mid \epsilon$$

Como por norma las gramáticas de tipo 3 no son ambiguas podemos decir que esta al serlo no puede ser ambigua.

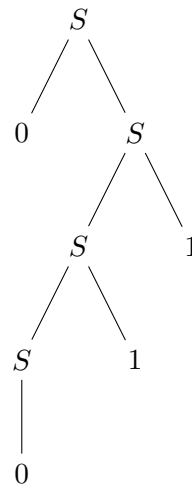
#### 4.1.2. Apartado B

$$S \rightarrow 0S1 \mid S1 \mid 0S \mid 0$$

Palabra 0011



Palabra 0011



Como tenemos dos grafos que dan la misma palabra podemos decir que la gramática es ambigua. Y como podemos encontrar la gramática  $S \rightarrow S1|S0|0$ , diremos también que es inherentemente ambigua.

### 4.1.3. Apartado C

$$S \rightarrow A1B \qquad A \rightarrow 0A \mid \epsilon \qquad B \rightarrow 0B \mid 1B \mid \epsilon$$

Como podemos ver a simple vista, las S generan A por la izquierda y B por la derecha por lo que nunca podremos encontrar dos arboles iguales. Esto nos dice que la gramática no es ambigua y por lo tanto el lenguaje no puede ser inherentemente ambiguo.

## 4.2. Ejercicio 2

*Eliminar símbolos y producciones inútiles. Realizar el procedimiento paso por paso, indicando las variables descartadas y el motivo.*

$$\begin{array}{llll} S \rightarrow moA & S \rightarrow cI & A \rightarrow dEs & A \rightarrow jBI \\ B \rightarrow bb & B \rightarrow D & E \rightarrow elO & E \rightarrow Perl \\ D \rightarrow de & C \rightarrow c & J \rightarrow kC & I \rightarrow fl \\ O \rightarrow o & P \rightarrow ola & & \end{array}$$

En primer lugar, eliminaremos las variables que no llegan a las palabras  $T^*$  y las producciones donde estas estén:

$$V_t = \{C\}$$

$$\begin{array}{llll} S \rightarrow moA & S \rightarrow cI & A \rightarrow dEs & A \rightarrow jBI \\ B \rightarrow bb & B \rightarrow D & E \rightarrow elO & E \rightarrow Perl \\ D \rightarrow de & \color{blue}{C \rightarrow c} & J \rightarrow kC & I \rightarrow fl \\ O \rightarrow o & P \rightarrow ola & & \end{array}$$

$$V_t = \{C, O\}$$

$$S \rightarrow moA$$

$$S \rightarrow cI$$

$$A \rightarrow dEs$$

$$A \rightarrow jBI$$

$$B \rightarrow bb$$

$$B \rightarrow D$$

$$E \rightarrow elO$$

$$E \rightarrow Perl$$

$$D \rightarrow de$$

$$C \rightarrow c$$

$$J \rightarrow kC$$

$$I \rightarrow fl$$

$$O \rightarrow o$$

$$P \rightarrow ola$$

$$V_t = \{C, O, B\}$$

$$S \rightarrow moA$$

$$S \rightarrow cI$$

$$A \rightarrow dEs$$

$$A \rightarrow jBI$$

$$B \rightarrow bb$$

$$B \rightarrow D$$

$$E \rightarrow elO$$

$$E \rightarrow Perl$$

$$D \rightarrow de$$

$$C \rightarrow c$$

$$J \rightarrow kC$$

$$I \rightarrow fl$$

$$O \rightarrow o$$

$$P \rightarrow ola$$

$$V_t = \{C, O, B, D\}$$

$$S \rightarrow moA$$

$$S \rightarrow cI$$

$$A \rightarrow dEs$$

$$A \rightarrow jBI$$

$$B \rightarrow bb$$

$$B \rightarrow D$$

$$E \rightarrow elO$$

$$E \rightarrow Perl$$

$$D \rightarrow de$$

$$C \rightarrow c$$

$$J \rightarrow kC$$

$$I \rightarrow fl$$

$$O \rightarrow o$$

$$P \rightarrow ola$$

$$V_t = \{C, O, B, D, J\}$$

$$S \rightarrow moA$$

$$S \rightarrow cI$$

$$A \rightarrow dEs$$

$$A \rightarrow jBI$$

$$B \rightarrow bb$$

$$B \rightarrow D$$

$$E \rightarrow elO$$

$$E \rightarrow Perl$$

$$D \rightarrow de$$

$$C \rightarrow c$$

$$J \rightarrow kC$$

$$I \rightarrow fl$$

$$O \rightarrow o$$

$$P \rightarrow ola$$

$$V_t = \{C, O, B, D, J, E\}$$

$$\begin{array}{llll}
S \rightarrow moA & S \rightarrow cI & A \rightarrow dEs & A \rightarrow jBI \\
B \rightarrow bb & B \rightarrow D & E \rightarrow elO & E \rightarrow Perl \\
D \rightarrow de & C \rightarrow c & J \rightarrow kC & I \rightarrow fl \\
O \rightarrow o & P \rightarrow ola & & 
\end{array}$$

$$V_t = \{C, O, B, D, J, E, A\}$$

$$\begin{array}{llll}
S \rightarrow moA & S \rightarrow cI & A \rightarrow dEs & A \rightarrow jBI \\
B \rightarrow bb & B \rightarrow D & E \rightarrow elO & E \rightarrow Perl \\
D \rightarrow de & C \rightarrow c & J \rightarrow kC & I \rightarrow fl \\
O \rightarrow o & P \rightarrow ola & & 
\end{array}$$

$$V_t = \{C, O, B, D, J, E, A, S\}$$

$$\begin{array}{llll}
S \rightarrow moA & S \rightarrow cI & A \rightarrow dEs & A \rightarrow jBI \\
B \rightarrow bb & B \rightarrow D & E \rightarrow elO & E \rightarrow Perl \\
D \rightarrow de & C \rightarrow c & J \rightarrow kC & I \rightarrow fl \\
O \rightarrow o & P \rightarrow ola & & 
\end{array}$$

$$V_{resultado} = V - V_t = \{P, I\}$$

Ahora continuamos eliminando los estados que no se pueden alcanzar desde el estado inicial S, y las producciones que estos tienen:

$$\{ v_S: \text{variables obtenidas} \quad T_S: \text{símbolos terminales} \quad J: \text{variables por analizar} \{S\} \}$$

$$\begin{array}{llll}
S \rightarrow moA & S \rightarrow cI & A \rightarrow dEs & A \rightarrow jBI \\
B \rightarrow bb & B \rightarrow D & E \rightarrow elO & E \rightarrow Perl \\
D \rightarrow de & C \rightarrow c & J \rightarrow kC & O \rightarrow o \\
\{ v_S: \{ S \} & T_S: \{ m,o \} & J: \{ A \} \} & 
\end{array}$$

$$\begin{array}{llll}
S \rightarrow moA & S \rightarrow cI & A \rightarrow dEs & A \rightarrow jBI \\
B \rightarrow bb & B \rightarrow D & E \rightarrow elO & E \rightarrow Perl \\
D \rightarrow de & C \rightarrow c & J \rightarrow kC & O \rightarrow o \\
\{ v_S: \{ S,A \} & T_S: \{ m,o,d,s \} & J: \{ E \} \} & 
\end{array}$$

$$\begin{array}{llll}
S \rightarrow moA & S \rightarrow cI & A \rightarrow dEs & A \rightarrow jBI \\
B \rightarrow bb & B \rightarrow D & E \rightarrow elO & E \rightarrow Perl \\
D \rightarrow de & C \rightarrow c & J \rightarrow kC & O \rightarrow o \\
\{ v_S: \{ S,A,E \} & T_S: \{ m,o,d,s,e,l \} & J: \{ O \} \} & 
\end{array}$$

$$\begin{array}{llll}
S \rightarrow moA & S \rightarrow cI & A \rightarrow dEs & A \rightarrow jBI \\
B \rightarrow bb & B \rightarrow D & E \rightarrow elO & E \rightarrow Perl \\
D \rightarrow de & C \rightarrow c & J \rightarrow kC & O \rightarrow o \\
\{ v_S: \{ S,A,E,O \} & T_S: \{ m,o,d,s,e,l,o \} & J: \{ \} \} & 
\end{array}$$

Por lo tanto las producciones que nos quedan son:

$$S \rightarrow moA \quad A \rightarrow dEs \quad E \rightarrow elO \quad O \rightarrow o$$

Estas producciones proporcionan la siguiente palabra:

$$S \rightarrow moA \Rightarrow modEs \Rightarrow modelOs \Rightarrow \text{modelos}$$

### 4.3. Ejercicio 3

Eliminar producciones nulas y unitarias, en el orden correcto. Realizar los procedimientos paso por paso, indicando las producciones descartadas en cada momento

$$\begin{array}{llll}
 S \rightarrow XYZ & S \rightarrow XYz & X \rightarrow xxX & X \rightarrow \epsilon \\
 Y \rightarrow yyY & Y \rightarrow \epsilon & Z \rightarrow yxZ & Z \rightarrow X
 \end{array}$$

En el primer paso **quitamos las producciones nulas**.

$$\begin{array}{llll}
 S \rightarrow XYZ & S \rightarrow XYz & X \rightarrow xxX & \textcolor{red}{X \rightarrow \epsilon} \\
 Y \rightarrow yyY & \textcolor{red}{Y \rightarrow \epsilon} & Z \rightarrow yxZ & Z \rightarrow X
 \end{array}$$

Vemos que variables son anulables para extraerlas, y obtenemos que:

$$\begin{array}{l}
 X \rightarrow \epsilon \\
 Z \rightarrow X \rightarrow \epsilon \\
 Y \rightarrow \epsilon \\
 S \rightarrow XYZ \rightarrow \epsilon
 \end{array}$$

Teniendo en cuenta que S es anulable la palabra vacía podría crearse con esta gramática.  
 $H = \{X, Y, Z, S\}$

Parte dos del proceso, **añadir producciones**.

En la tabla señalamos en **rojo** la parte que se elimina o no se edita y quedaran fuera en el futuro:

	$S \rightarrow XYZ$	$X \rightarrow xxX$	$Y \rightarrow yyY$	$S \rightarrow XYz$	$Z \rightarrow yxZ$	$Z \rightarrow X$
delet X	$S \rightarrow YZ\textcolor{red}{X}$	$X \rightarrow xx\textcolor{red}{X}$	$Y \rightarrow yy\textcolor{red}{Y}$	$S \rightarrow \textcolor{red}{X}Yz$	$Z \rightarrow yx\textcolor{red}{Z}$	$Z \rightarrow \textcolor{red}{X}$
delet Y	$S \rightarrow X\textcolor{red}{Y}Z$	$X \rightarrow xx\textcolor{red}{X}$	$Y \rightarrow yy\textcolor{red}{Y}$	$S \rightarrow X\textcolor{red}{Y}z$	$Z \rightarrow yx\textcolor{red}{Z}$	$Z \rightarrow \textcolor{red}{X}$
delet Z	$S \rightarrow XY\textcolor{red}{Z}$	$X \rightarrow xx\textcolor{red}{X}$	$Y \rightarrow yy\textcolor{red}{Y}$	$S \rightarrow XY\textcolor{red}{z}$	$Z \rightarrow yx\textcolor{red}{Z}$	$Z \rightarrow \textcolor{red}{X}$
delet XY	$S \rightarrow \textcolor{red}{X}Y\textcolor{red}{Z}$	$X \rightarrow xx\textcolor{red}{X}$	$Y \rightarrow yy\textcolor{red}{Y}$	$S \rightarrow \textcolor{red}{X}Y\textcolor{red}{z}$	$Z \rightarrow yx\textcolor{red}{Z}$	$Z \rightarrow \textcolor{red}{X}$
delet YZ	$S \rightarrow X\textcolor{red}{Y}Z$	$X \rightarrow xx\textcolor{red}{X}$	$Y \rightarrow yy\textcolor{red}{Y}$	$S \rightarrow X\textcolor{red}{Y}z$	$Z \rightarrow yx\textcolor{red}{Z}$	$Z \rightarrow \textcolor{red}{X}$
delet XZ	$S \rightarrow \textcolor{red}{X}Y\textcolor{red}{Z}$	$X \rightarrow xx\textcolor{red}{X}$	$Y \rightarrow yy\textcolor{red}{Y}$	$S \rightarrow \textcolor{red}{X}Yz$	$Z \rightarrow yx\textcolor{red}{Z}$	$Z \rightarrow \textcolor{red}{X}$

Por lo tanto limpiando los eliminados tendríamos los siguientes valores:

	$S \rightarrow XYZ$	$X \rightarrow xxX$	$Y \rightarrow yyY$	$S \rightarrow XYz$	$Z \rightarrow yxZ$	$Z \rightarrow X$
delet X	$S \rightarrow YZ$	$X \rightarrow xx$		$S \rightarrow Yz$		
delet Y	$S \rightarrow XZ$		$Y \rightarrow yy$	$S \rightarrow Xz$		
delet Z	$S \rightarrow XY$				$Z \rightarrow yx$	
delet XY	$S \rightarrow Z$	$X \rightarrow xx$	$Y \rightarrow yy$	$S \rightarrow z$		
delet YZ	$S \rightarrow X$		$Y \rightarrow yy$	$S \rightarrow Xz$	$Z \rightarrow yx$	
delet XZ	$S \rightarrow Y$	$X \rightarrow xx$		$S \rightarrow Yz$	$Z \rightarrow yx$	

Eliminamos los repetidos que son los de color azul:

	$S \rightarrow XYZ$	$X \rightarrow xxX$	$Y \rightarrow yyY$	$S \rightarrow XYz$	$Z \rightarrow yxZ$	$Z \rightarrow X$
delet X	$S \rightarrow YZ$	$X \rightarrow xx$		$S \rightarrow Yz$		
delet Y	$S \rightarrow XZ$		$Y \rightarrow yy$	$S \rightarrow Xz$		
delet Z	$S \rightarrow XY$				$Z \rightarrow yx$	
delet XY	$S \rightarrow Z$	$X \rightarrow xx$	$Y \rightarrow yy$	$S \rightarrow z$		
delet YZ	$S \rightarrow X$		$Y \rightarrow yy$	$S \rightarrow Xz$	$Z \rightarrow yx$	
delet XZ	$S \rightarrow Y$	$X \rightarrow xx$		$S \rightarrow Yz$	$Z \rightarrow yx$	

Limpiamos la tabla y obtenemos los siguientes valores:

	$S \rightarrow XYZ$	$X \rightarrow xxX$	$Y \rightarrow yyY$	$S \rightarrow XYz$	$Z \rightarrow yxZ$	$Z \rightarrow X$
delet X	$S \rightarrow YZ$	$X \rightarrow xx$		$S \rightarrow Yz$		
delet Y	$S \rightarrow XZ$		$Y \rightarrow yy$	$S \rightarrow Xz$		
delet Z	$S \rightarrow XY$				$Z \rightarrow yx$	
delet XY	$S \rightarrow Z$			$S \rightarrow z$		
delet YZ	$S \rightarrow X$					
delet XZ	$S \rightarrow Y$					

El siguiente paso es eliminar las unidades unitarias. Como tenemos que:

$$S \rightarrow Z \quad S \rightarrow Y \quad S \rightarrow X \quad Z \rightarrow X$$

Como podemos ver, Z está a la derecha y a la izquierda por lo tanto tendríamos que añadir  $(S, X)$  pero como ya está no es necesario, ni necesitamos meterla. Ya que si luego tendríamos que eliminarla cuando limpiemos.



Por lo que al final nos queda que: { (S,X) (S,Y) (S,Z) (Z,X) }

$S \rightarrow XYZ$	$X \rightarrow xxX$	$Y \rightarrow yyY$	$S \rightarrow XYz$	$Z \rightarrow yxZ$
$S \rightarrow YZ$	$X \rightarrow xx$		$S \rightarrow Yz$	
$S \rightarrow XZ$		$Y \rightarrow yy$	$S \rightarrow Xz$	
$S \rightarrow XY$				$Z \rightarrow yx$
	$S \rightarrow xxX$	$S \rightarrow yyY$		$S \rightarrow yxZ$
	$S \rightarrow xx$	$S \rightarrow yy$		$S \rightarrow yx$
	$Z \rightarrow xxX$			
	$Z \rightarrow xx$			

#### 4.4. Ejercicio 4

Pasar la siguiente gramática a forma normal de Greibach:

$$S \rightarrow a \mid CD \mid CS \quad A \rightarrow a \mid b \mid SS \quad C \rightarrow a \quad D \rightarrow AS$$

Primero renombramos las variables quedando lo siguiente:

$$S = A_1 \quad C = A_2 \quad D = A_3 \quad A = A_4$$

$$\begin{aligned} A_1 &\rightarrow a \mid A_2A_3 \mid A_2A_1 & A_2 &\rightarrow a & A_3 &\rightarrow A_4A_1 \\ A_4 &\rightarrow a \mid b \mid A_1A_1 \end{aligned}$$

Ahora continuamos con la transformación pre-Greibach, para ello nos fijamos que todas las producciones han de estar de la forma  $i < j$ .

$$\begin{aligned} A_4 &\rightarrow A_1A_1 \Rightarrow A_4 \rightarrow aA_1 \\ A_4 &\rightarrow A_2A_3A_1 \Rightarrow A_4 \rightarrow aA_3A_1 \\ A_4 &\rightarrow A_2A_1A_1 \Rightarrow A_4 \rightarrow aA_1A_1 \end{aligned}$$

$$\begin{aligned} A_1 &\rightarrow a \mid A_2A_3 \mid A_2A_1 & A_2 &\rightarrow a & A_3 &\rightarrow A_4A_1 \\ A_4 &\rightarrow a \mid b \mid aA_1 \mid aA_3A_1 \mid aA_1A_1 \end{aligned}$$

Ahora pasamos todas las formas formales de Greibach.

$$\begin{aligned} A_3 &\rightarrow A_4A_1 \\ \Rightarrow A_3 &\rightarrow aA_1 \Rightarrow A_3 \rightarrow bA_1 \Rightarrow A_3 \rightarrow aA_1A_1 \Rightarrow A_3 \rightarrow aA_3A_1A_1 \Rightarrow A_3 \rightarrow aA_1A_1A_1 \end{aligned}$$

$$\begin{aligned} A_3 &\rightarrow A_2A_3 \\ \Rightarrow A_1 &\rightarrow aA_3 \end{aligned}$$

$$\begin{aligned}
A_1 &\rightarrow A_2 A_1 \\
&\Rightarrow A_1 \rightarrow a A_1
\end{aligned}$$

Entonces nuestra gramática quedaría de la siguiente forma:

$$\begin{aligned}
A_1 &\rightarrow a \mid a A_3 \mid a A_1 \\
A_2 &\rightarrow a \\
A_3 &\rightarrow a A_1 \mid b A_1 \mid a A_1 A_1 \mid a A_3 A_1 A_1 \mid a A_1 A_1 A_1 \\
A_4 &\rightarrow a \mid b \mid a A_1 \mid a A_3 A_1 \mid a A_1 A_1
\end{aligned}$$