



# PRÁCTICA 2

Israel Fargas Asquith  
Miriam Rodríguez Franco  
Marina Vizcaíno Bayo

# ÍNDICE

<b>ÍNDICE</b>	<b>1</b>
<b>1.- Introducción</b>	<b>2</b>
1.1.- Marco teórico	2
1.2.- Autómata Finito Determinista	2
1.2.1.- Funcionamiento del autómata	3
2.1.- Autómata Finito No Determinista	10
2.2.1.- Funcionamiento del autómata	10
<b>Bibliografía</b>	<b>14</b>

# 1.- Introducción

## 1.1.- Marco teórico

Los autómatas se usan para identificar cadenas de lenguajes formales de tipo 3 según la categoría de Chomsky. Dentro de los autómatas hay varias clases, nosotros estudiaremos los más simples que pueden ser de Máquinas Secuenciales (**Autómatas Finitos Deterministas**, **Autómatas Finitos No Deterministas**, Autómatas Probabilísticos) Máquinas Células de McCulloch-Pitts. Para cada gramática regular existe al menos un AFND que la cumple, y para cada AFND hay un AFD que la expresa de forma más sencilla computacionalmente pero más difícil de comprender al principio. Cabe decir que el uso que se da a estos autómatas es diario, sobre todo en el ámbito de la programación ya que son esenciales para desarrollar y estudiar los compiladores. El primer ejemplo implementable de su uso es un autómata que permite reconocer que variables pueden o no ser declaradas en un lenguaje.

Nuestra práctica se divide en dos partes. La primera parte trata sobre Autómatas Finitos Deterministas (AFD), y la segunda parte se enfoca en Autómatas Finitos No Deterministas.

## 1.2.- Autómata Finito Determinista

Un autómata se considera determinista cuando, para cada símbolo de entrada, existe únicamente un estado válido. Nuestra práctica implica la creación de un autómata y un lenguaje asociado. En este contexto, el autómata recibirá una cadena como entrada y deberá verificar si dicha cadena es válida o no.

Su definición formal es la siguiente:

Un autómata finito determinista consiste en una **quíntupla** (S, E, P, I, F) donde:

**S**: Es un conjunto finito de estados.

**E**: Es el alfabeto del autómata.

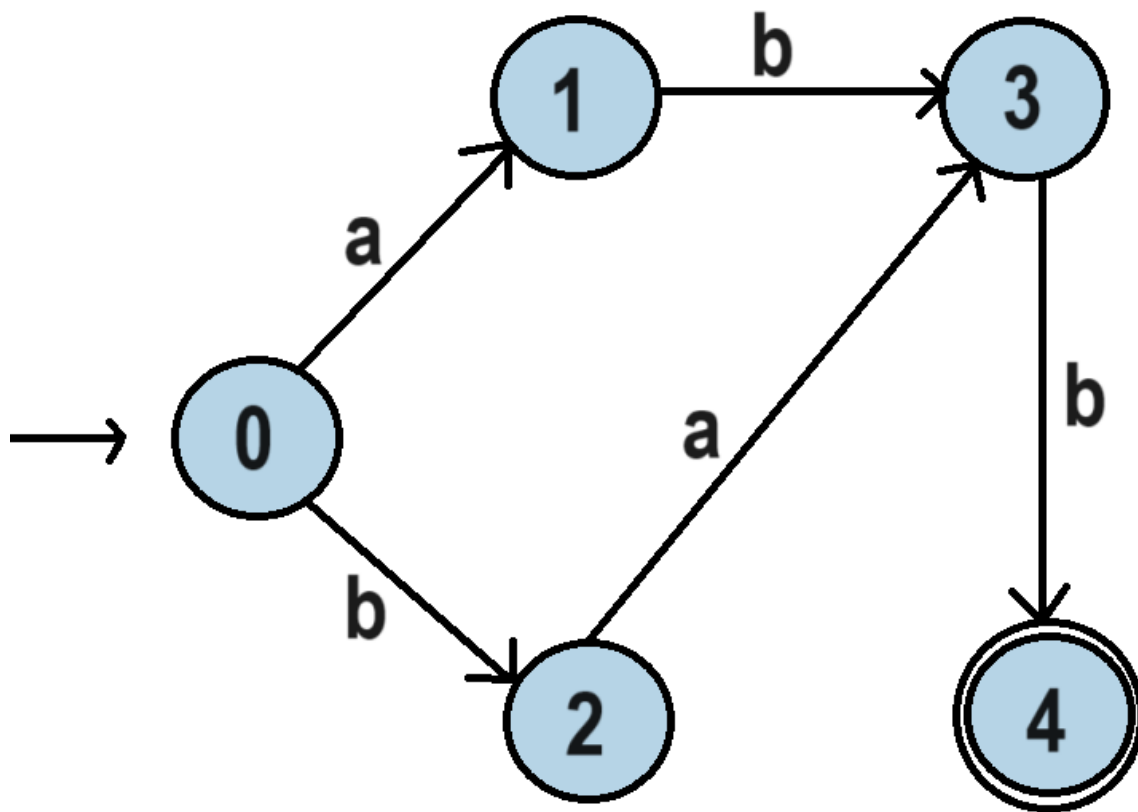
**P**: Es la función de transición de  $S \times E$  a  $S$ .

**I**: Es un elemento de  $S$  que es el estado inicial.

**F**: Es un subconjunto de  $S$  que componen los estados finales.

La función de transición se define en los AFD de forma que la máquina pueda pasar de un estado  $q_i$  a un estado  $q_j$  por medio de solo una transición para cualesquiera valores  $i, j$ . Incluyendo  $i = j$ .

## 1.2.1.- Funcionamiento del autómata



Para este AFD realizaremos las siguiente pruebas:

Introduce el número de nodos: 5

Introduce símbolos: a b

Estado Inicial: 0

Estado Final: 4

Transiciones:

- 0 'a' 1
- 0 'b' 2
- 1 'b' 3
- 2 'a' 3
- 3 'b' 4

Introducir Cadena a comprobar: abb

Volver OK

salida:

```
0 'a' 1
0 'b' 2
1 'b' 3
2 'a' 3
3 'b' 4
Estado: 0      SIMBOLO: a
Estado: 1      SIMBOLO: b
Estado: 3      SIMBOLO: b
Es final
```

Como podemos comprobar la cadena acaba en un estado final, por lo tanto es una cadena válida.

A screenshot of a graphical user interface for configuring a finite automaton. The window has a title bar with standard minimize, maximize, and close buttons. The main area contains several input fields and a list box. The labels and their corresponding values are: 'Introduce el número de nodos' with '5', 'Introduce símbolos' with 'a b', 'Estado Inicial' with '0', 'Estado Final' with '4', and 'Transiciones' with a list box containing five entries: '0 'a' 1', '0 'b' 2', '1 'b' 3', '2 'a' 3', and '3 'b' 4'. At the bottom, there is an 'Introducir Cadena a comprobar' field with 'bab' and two buttons labeled 'Volver' and 'OK'.

Label	Value
Introduce el número de nodos	5
Introduce símbolos	a b
Estado Inicial	0
Estado Final	4
Transiciones	0 'a' 1 0 'b' 2 1 'b' 3 2 'a' 3 3 'b' 4
Introducir Cadena a comprobar	bab

```
0 'a' 1
0 'b' 2
1 'b' 3
2 'a' 3
3 'b' 4
Estado: 0      SIMBOLO: b
Estado: 2      SIMBOLO: a
Estado: 3      SIMBOLO: b
Es final
```

Como podemos comprobar la cadena acaba en un estado final, por lo tanto es una cadena válida.

A screenshot of a graphical user interface for configuring a finite automaton. The window has a title bar with a small icon and standard window controls (minimize, maximize, close). The main area contains several input fields and a list box, each with a label to its left:

- Introduce el número de nodos**: A text box containing the number '5'.
- Introduce símbolos**: A text box containing the string 'a b'.
- Estado Inicial**: A text box containing the number '0'.
- Estado Final**: A text box containing the number '4'.
- Transiciones**: A list box containing the following text:

```
0 'a' 1
0 'b' 2
1 'b' 3
2 'a' 3
3 'b' 4
```
- Introducir Cadena a comprobar**: A text box containing the string 'a'.

At the bottom right of the window are two buttons: 'Volver' and 'OK'.

```
0 'a' 1
```

```
0 'b' 2
```

```
1 'b' 3
```

```
2 'a' 3
```

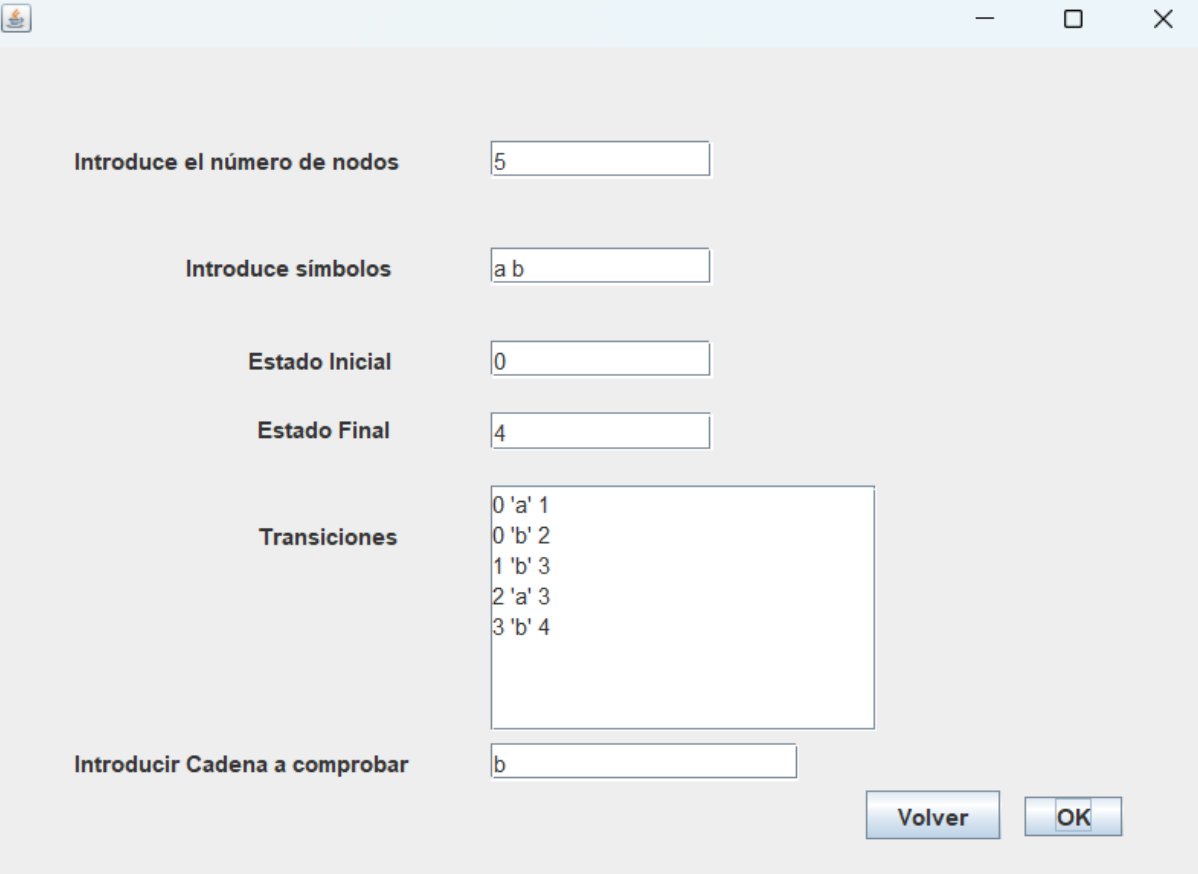
```
3 'b' 4
```

```
Estado: 0
```

```
SIMBOLO: a
```

```
No es final
```

Como podemos comprobar la cadena no acaba en un estado final, por lo tanto no es una cadena válida.



A screenshot of a graphical user interface for configuring a finite automaton. The window has a title bar with standard OS controls. The main area contains several input fields and a list box. The labels and their corresponding values are: 'Introduce el número de nodos' with '5', 'Introduce símbolos' with 'a b', 'Estado Inicial' with '0', 'Estado Final' with '4', and 'Introducir Cadena a comprobar' with 'b'. The 'Transiciones' section contains a list box with five entries: '0 'a' 1', '0 'b' 2', '1 'b' 3', '2 'a' 3', and '3 'b' 4'. At the bottom right, there are two buttons labeled 'Volver' and 'OK'.

Label	Value
Introduce el número de nodos	5
Introduce símbolos	a b
Estado Inicial	0
Estado Final	4
Transiciones	0 'a' 1 0 'b' 2 1 'b' 3 2 'a' 3 3 'b' 4
Introducir Cadena a comprobar	b

Volver OK

```
0 'a' 1
0 'b' 2
1 'b' 3
2 'a' 3
3 'b' 4
Estado: 0          SIMBOLO: b
· No es final
```

Como podemos comprobar la cadena no acaba en un estado final, por lo tanto no es una cadena válida.



Introduce el número de nodos: 5

Introduce símbolos: a b

Estado Inicial: 0

Estado Final: 4

Transiciones:

- 0 'a' 1
- 0 'b' 2
- 1 'b' 3
- 2 'a' 3
- 3 'b' 4

Introducir Cadena a comprobar: ab

Volver OK

0 'a' 1

0 'b' 2

1 'b' 3

2 'a' 3

3 'b' 4

Estado: 0                      SIMBOLO: a

Estado: 1                      SIMBOLO: b

No es final

Como podemos comprobar la cadena no acaba en un estado final, por lo tanto no es una cadena válida.

The screenshot shows a graphical user interface for building a finite automaton. It includes several input fields and a list of transitions.

Field Label	Value
Introduce el número de nodos	5
Introduce símbolos	a b
Estado Inicial	0
Estado Final	4
Transiciones	0 'a' 1 0 'b' 2 1 'b' 3 2 'a' 3 3 'b' 4
Introducir Cadena a comprobar	ba

Buttons: Volver, OK

```
0 'a' 1
0 'b' 2
1 'b' 3
2 'a' 3
3 'b' 4
```

```
Estado: 0          SIMBOLO: b
```

```
Estado: 2          SIMBOLO: a
```

```
No es final
```

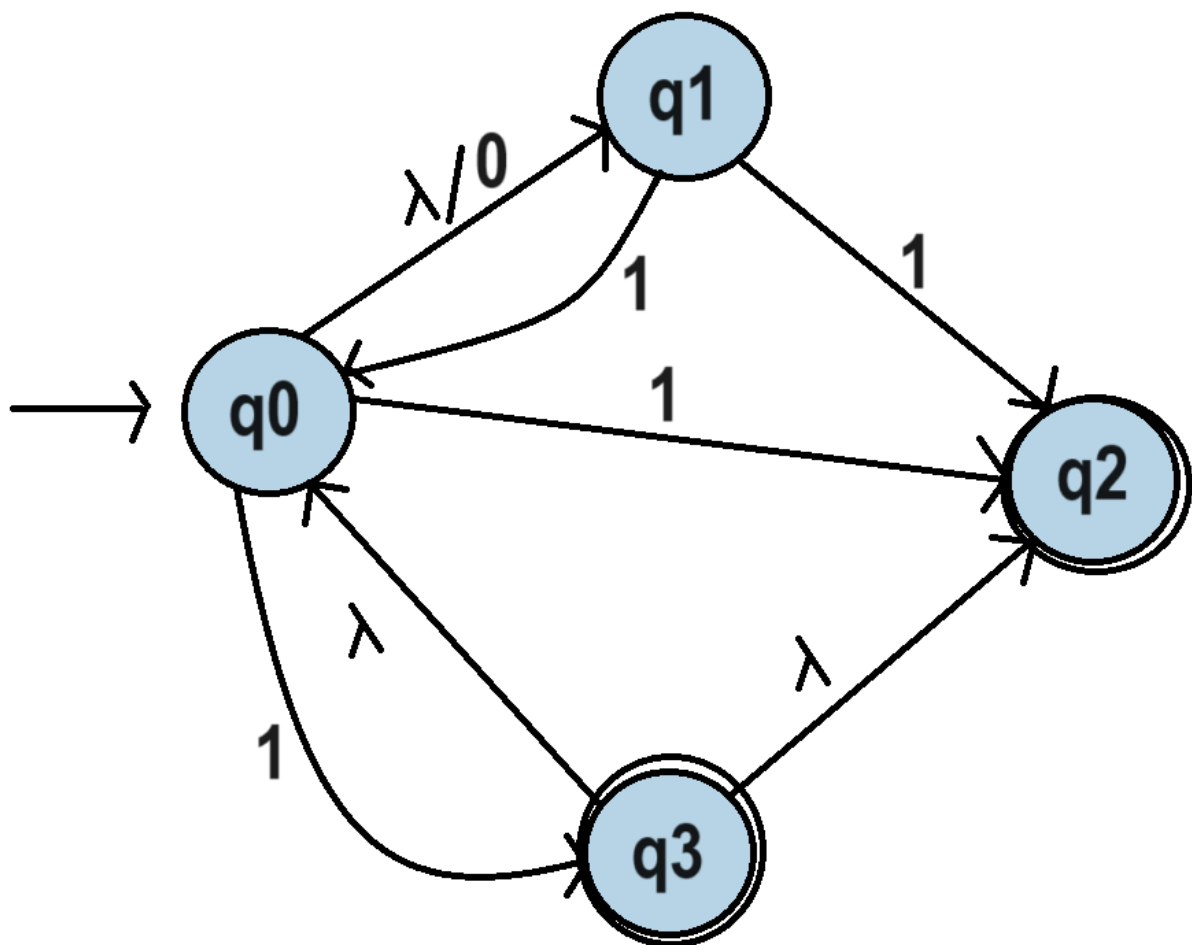
Como podemos comprobar la cadena no acaba en un estado final, por lo tanto no es una cadena válida.

## 2.1.- Autómata Finito No Determinista

La diferencia entre un autómata finito no determinista y uno determinista es que la función de transición **P** permite la multiplicidad de transiciones desde un estado a otros con el mismo símbolo. Además implementa el carácter Lambda para cualquier gramática, dicho carácter no consume ninguno de la cadena a analizar.

Esto puede parecer muy complejo pero es la forma más sencilla de desarrollar un autómata para adaptarse a cierto lenguaje. El problema es que la implementación debe tener de forma obligatoria algún algoritmo de backtracking o similar para analizar todos los múltiples caminos posibles.

### 2.2.1.- Funcionamiento del autómata



Para este AFDN realizaremos las siguientes pruebas:

Introduce el número de nodos: 4

Introduce símbolos: 0 1

Estado Inicial: q0

Estado Final: q2 q3

Transiciones:  
q0 '0' q1  
q0 '1' q2 q3  
q1 '1' q0 q2

Transiciones LAMBDA:  
q0 q1  
q3 q0 q2

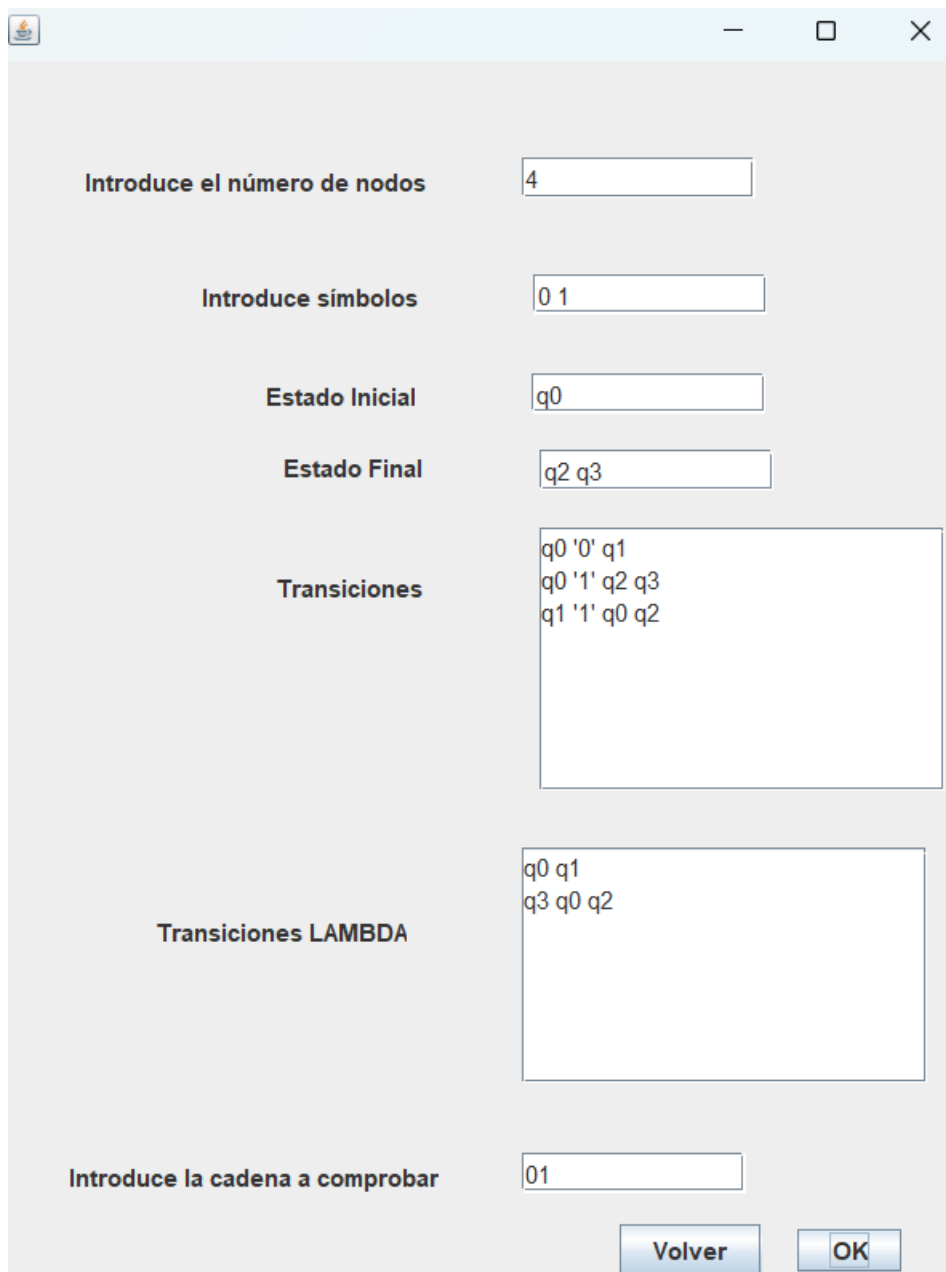
Introduce la cadena a comprobar: 1

Volver OK

Salida

```
q0 '0' q1
q0 '1' q2 q3
q1 '1' q0 q2
q0 q1
q3 q0 q2
Es final
```

Como podemos comprobar  
acaba en estado final  
por lo tanto es una cadena  
válida



A screenshot of a software window for configuring a finite automaton. The window has a title bar with standard OS controls. It contains several input fields and text areas for defining the automaton's components.

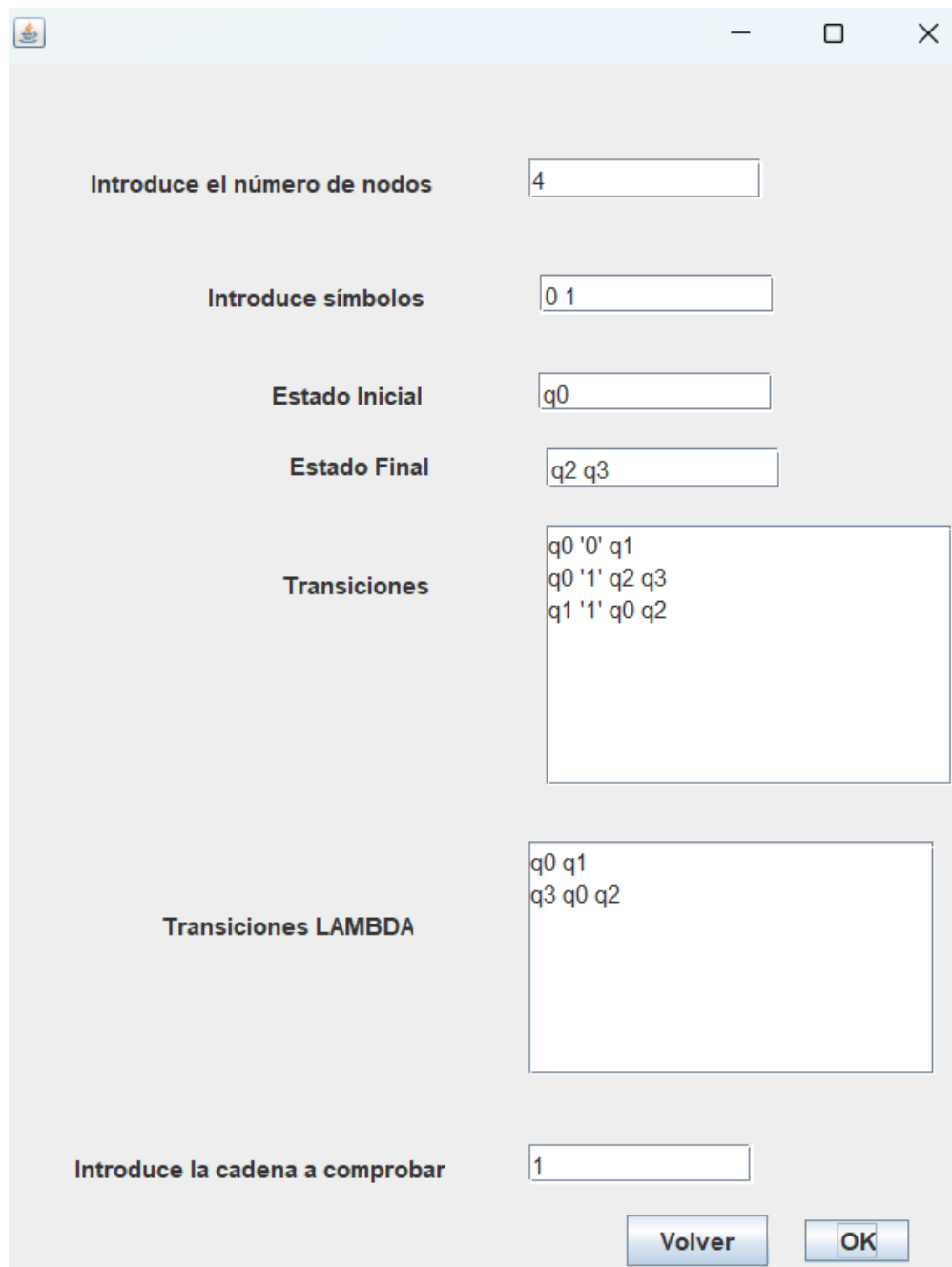
Field Label	Value
Introduce el número de nodos	4
Introduce símbolos	0 1
Estado Inicial	q0
Estado Final	q2 q3
Transiciones	q0 '0' q1 q0 '1' q2 q3 q1 '1' q0 q2
Transiciones LAMBDA	q0 q1 q3 q0 q2
Introduce la cadena a comprobar	01

At the bottom right, there are two buttons: "Volver" and "OK".

salida:

```
q0 '0' q1
q0 '1' q2 q3
q1 '1' q0 q2
q0 q1
q3 q0 q2
Es final
```

Como podemos  
comprobar acaba en  
estado final por lo tanto  
es una cadena válida



A screenshot of a graphical user interface for configuring a finite automaton. The window has a title bar with standard OS controls. It contains several input fields and text areas for defining the automaton's components.

Field Label	Value
Introduce el número de nodos	4
Introduce símbolos	0 1
Estado Inicial	q0
Estado Final	q2 q3
Transiciones	q0 '0' q1 q0 '1' q2 q3 q1 '1' q0 q2
Transiciones LAMBDA	q0 q1 q3 q0 q2
Introduce la cadena a comprobar	1

At the bottom right, there are two buttons: "Volver" and "OK".

salida:

```
q0 '0' q1
q0 '1' q2 q3
q1 '1' q0 q2
q0 q1
q3 q0 q2
Es final
```

Como podemos comprobar acaba en estado final por lo tanto es una cadena válida

# Bibliografía

Teoría de la computación Lenguajes formales, autómatas y complejidad (J. Glenn  
Brookshear)

Lenguajes, Gramáticas y Autómatas un enfoque práctico (Pedro Isasi)