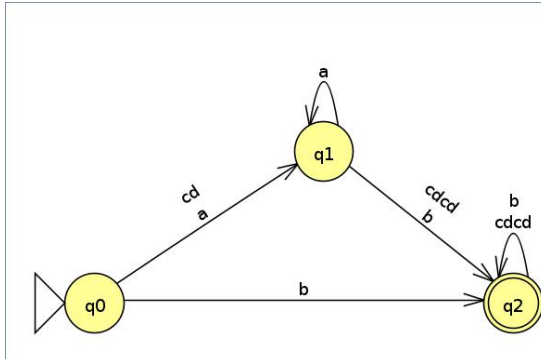


Práctico 3 - Autómatas Finitos y Traductores

Ejercicio 1.

Cuando sea posible, dar un autómata finito para los lenguajes de los ejercicios 1, 2, 3, 4 y 5 de la práctica 2.

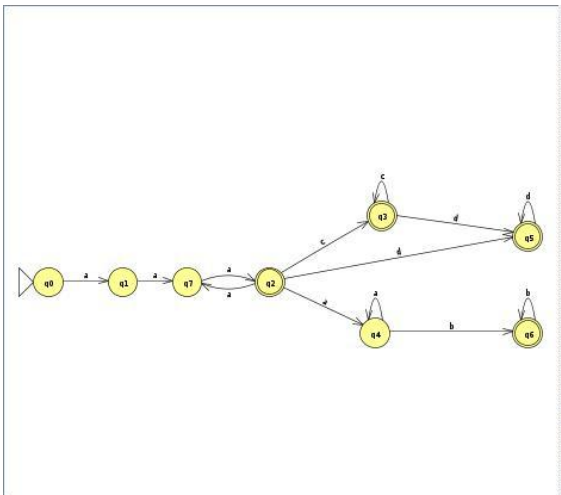
1a) $a^*b^+ \mid cd(cdcd)^+$



Input	Result
b	Accept
aa	Reject
ab	Accept
cdcdcd	Accept
cd	Reject
aab	Accept
cdcdcdcdcd	Accept
abb	Accept
cdcdcdcdcdcd	Accept
abbbbbbb	Accept

Load Inputs Run Inputs Clear Enter Lambda View Trace

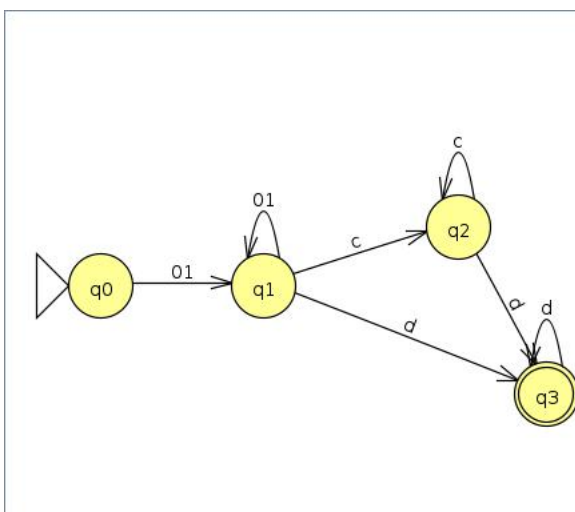
1b) $a(aa)^+ \cdot ((c^*d^*) \mid (ab)^+)$



Input	Result
aaacd	Accept
aaaaacdcd	Reject
aaaaaaa	Accept
aaaaaab	Accept
aaaaaaab	Accept
aaaaabcd	Reject
aab	Reject
aabcd	Reject
acd	Reject
ab	Reject
a	Reject
aaaaab	Accept
aaaaaab	Accept
aaaaacd	Accept
aaaaaab	Accept
aaacdcdcd	Reject
aaaabababab	Reject

Load Inputs Run Inputs Clear Enter Lambda View Trace

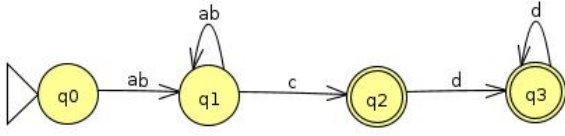
1c) $(01)^+ \cdot (c^*d^+)$



Input	Result
01d	
01cd	
0101dd	
01c	
01cccd	
01dc	
01010101c	
01010101cd	
01010101cdd	

Load Inputs Run Inputs Clear Enter Lambda View Trace

1d) $(ab)^+ \cdot c \cdot d^*$



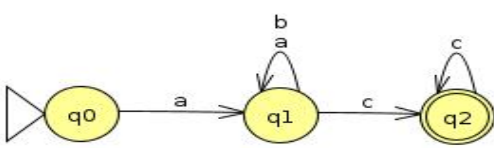
```

graph LR
    start(( )) --> q0((q0))
    q0 -- ab --> q1((q1))
    q1 -- ab --> q1
    q1 -- c --> q2(((q2)))
    q2 -- d --> q3(((q3)))
    q3 -- d --> q3
        
```

Input	Result
abc	
ababcd	
abcd	
abccdd	
abd	
abcc	

Load Inputs Run Inputs Clear Enter Lambda View Trac

1e) $a \cdot a^+ \cdot b^+ \cdot c^+$



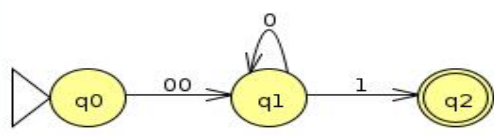
```

graph LR
    start(( )) --> q0((q0))
    q0 -- a --> q1((q1))
    q1 -- a --> q1
    q1 -- b --> q1
    q1 -- c --> q2(((q2)))
    q2 -- c --> q2
        
```

Input	Result
ac	Accept
acc	Accept
aac	Accept
aabcc	Accept
ac	Accept
bc	Reject
aaaaaabb	Reject
ab	Reject
abc	Accept
aaabbbccc	Accept

Load Inputs Run Inputs Clear Ente

1f) $0(0)^+ \cdot 1$



```

graph LR
    start(( )) --> q0((q0))
    q0 -- 00 --> q1((q1))
    q1 -- 0 --> q1
    q1 -- 1 --> q2(((q2)))
        
```

Input	Result
001	Accept
00001	Accept
0000001	Accept
01	Reject
010	Reject
0011	Reject

Load Inputs Run Inputs Clear Ente

1g) $(000 \cdot 1^* \cdot 0^*)^+$

```

graph LR
    start(( )) --> q0((q0))
    q0 -- 000 --> q1((q1))
    q1 -- 0 --> q1
    q1 -- 1 --> q1
    q1 -- 000 --> q1
    style start fill:none,stroke:none
    
```

Input	Result
000	Accept
0000	Accept
00010	Accept
000111	Accept
0001000	Accept
00	Reject
001	Reject
0001	Accept
10	Reject

1h) $aa(aa)^* \cdot (bbb)^*$

```

graph LR
    start(( )) --> q0((q0))
    q0 -- ab --> q1((q1))
    q1 -- ab --> q1
    q1 -- c --> q2((q2))
    q2 -- ba --> q3((q3))
    q3 -- baba --> q3
    style start fill:none,stroke:none
    
```

Input	Result
abcba	Accept
abcbababa	Accept
ababcbaba	Accept
ababcbabababa	Accept
abcc	Reject
abc	Reject
abcbaba	Reject
abcbababababa	Reject
abcb	Reject

1i) $(ab)^+ \cdot c \cdot ba(baba)^*$

```

graph LR
    start(( )) --> q0((q0))
    q0 -- aa --> q1((q1))
    q1 -- bbb --> q1
    q1 -- aa --> q1
    style start fill:none,stroke:none
    
```

Input	Result
aa	Accept
aaabbb	Accept
aaaabbbb	Accept
aaaa	Accept
aaaabbbbbbb	Accept
aaa	Reject
abbbb	Reject
bbb	Reject
aaabbbb	Reject
aab	Reject
aabb	Reject
bbaa	Reject

2a)

2b)

9
8
7
6
5
4
3
2
1
0

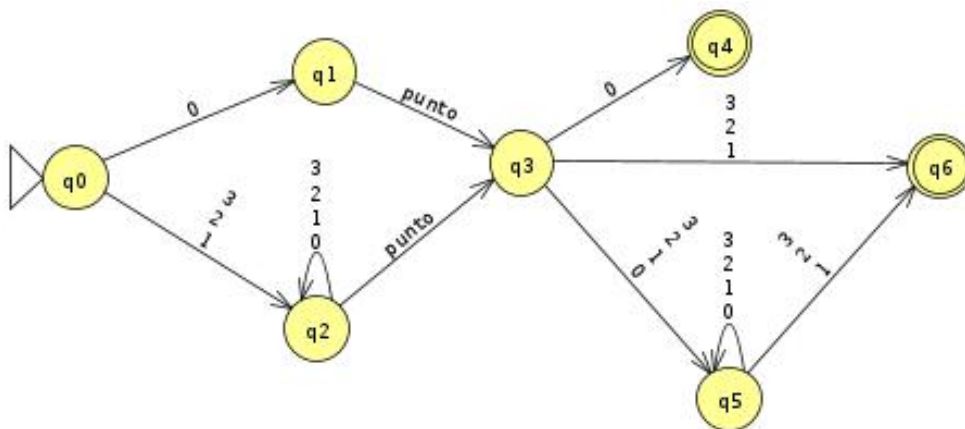
```

graph LR
    start(( )) --> q0((q0))
    q0 -- 0-9 --> q0
    q0 -- punto --> q1(((q1)))
    
```

Table Text Size

Input	Result
123456punto	Accept
0112233	Accept

2c)



2d)

2e)

2f)

3a) $(a(bc)^*)^+$.

```

graph LR
    start(( )) --> q0((q0))
    q0 -- a --> q1(((q1)))
    q1 -- a --> q1
    q1 -- bc --> q2(((q2)))
    
```

Input	Result
a	Accept
aa	Accept
abc	Accept
abcbc	Reject
aabca	Reject
abca	Reject
aaaa	Accept
abcbca	Reject

Load Inputs Run Inputs Clear Enter Lambda View Trace

3b) $(aaa)^*$

```

graph LR
    start(( )) --> q0(((q0)))
    q0 -- "λ  
aaa" --> q0
    
```

Input	Result
a	Reject
aaa	Accept
aaaaaa	Accept
ab	Reject
aaaa	Reject

Load Inputs Run Inputs Clear Enter Lambda View Trace

3c) $(b|c) \cdot (a|b|c)^*$

```

graph LR
    start(( )) --> q0((q0))
    q0 -- c --> q1(((q1)))
    q1 -- b --> q2(((q2)))
    q1 -- a --> q2
    q2 -- c --> q2
    q2 -- b --> q2
    q2 -- a --> q2
    
```

Input	Result
c	Accept
b	Accept
ca	Accept
cb	Accept
cc	Accept
ba	Accept
bb	Accept
bc	Accept
baa	Accept
bbb	Accept
bbc	Accept
ab	Reject
ac	Reject
caa	Accept

Load Inputs Run Inputs Clear Enter Lambda View Trace

3d) $(aaab)^*$

Input	Result
aaab	Accept
aaabaaab	Accept
aaabaaab	Accept
aaab	Reject
aaab	Reject
aaab	Reject
aaabaaab	Reject

4a) $0 \cdot 0 \cdot (0 / 1)^*$

Input	Result
00	Accept
000	Accept
0001	Accept
0010	Accept
001111111111	Accept
00000000001	Accept
01	Reject
001	Accept
001100	Accept
00110101	Accept

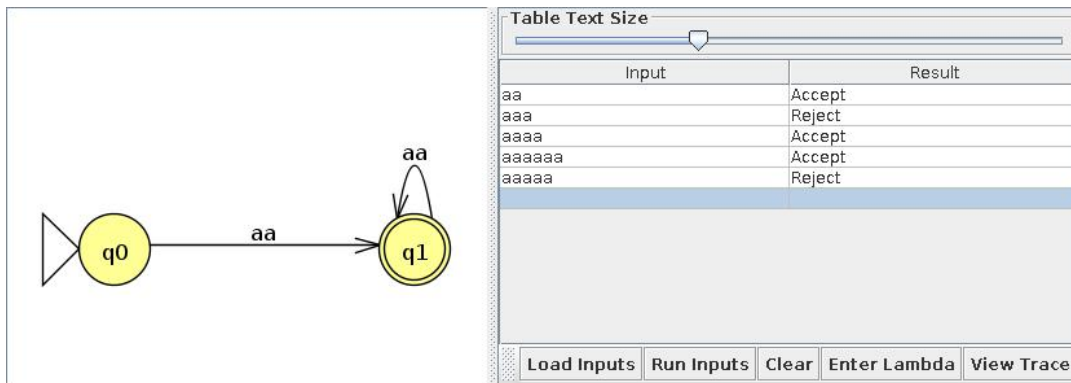
4b) $((0 \cdot 1) \mid 1)^*$

Input	Result
01	Accept
1	Accept
011	Accept
01101	Accept
101	Accept
110	Reject
11	Accept
1010	Reject
10	Reject

4c) $(a / b)^* \cdot (a \cdot a / b \cdot b)$

Input	Result
aa	Reject
bb	Accept
aaa	Accept
bbb	Accept
aaabbbabb	Accept
ba	Reject
ab	Reject
abba	Reject
baab	Reject
bbbb	Accept
bbb	Accept
bbaab	Reject

4d) $a \cdot a / (a \cdot a)^*$

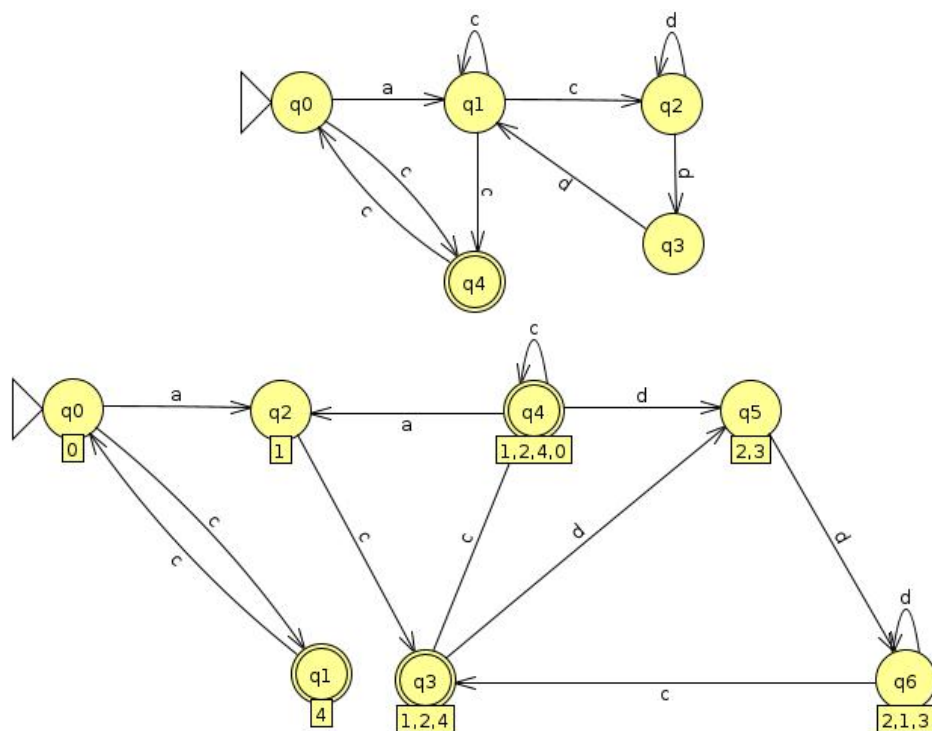


5a) $((0-2)(1-9)) \mid 3 (0-1) \text{ gui3n } 0(1-9) \mid 1(0-2) \text{ gui3n } (0-9)(0-9)(0-9)(0-9)$

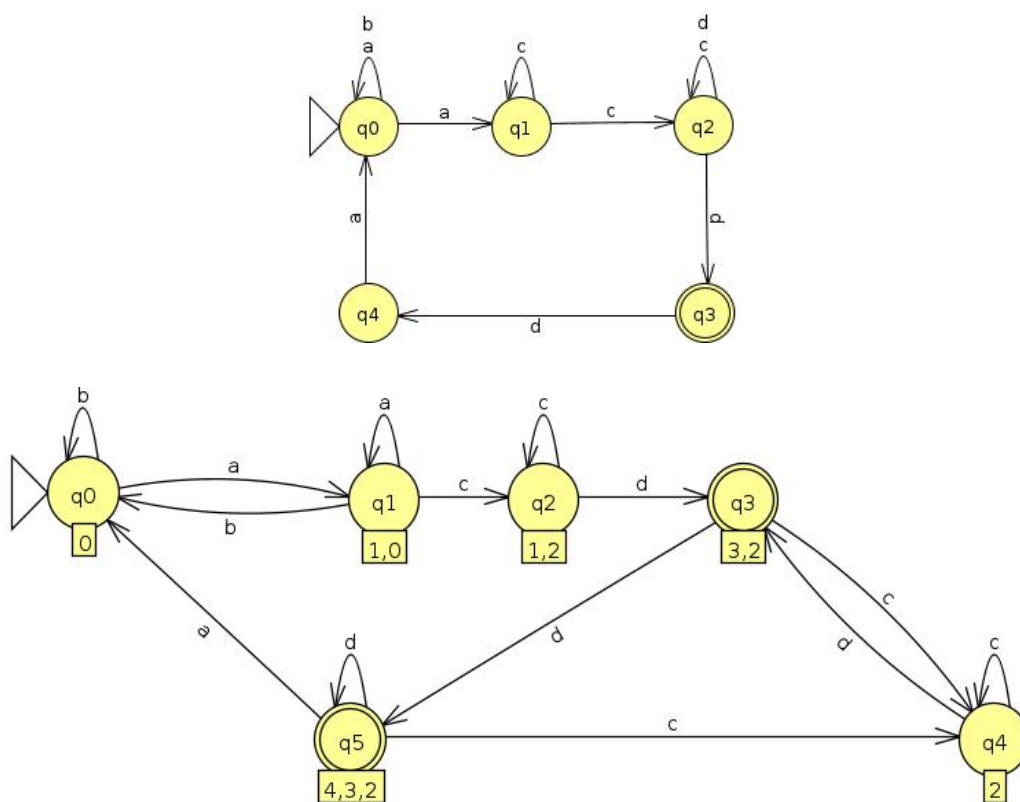
5b) $(0 \mid 1)(0-9) \mid 2(0-3) \text{ gui3n } (0-5)(0-9) \text{ gui3n } (0-5)(0-9)$

Ejercicio 2

a) AF original y AFD



b) AF original y AFD



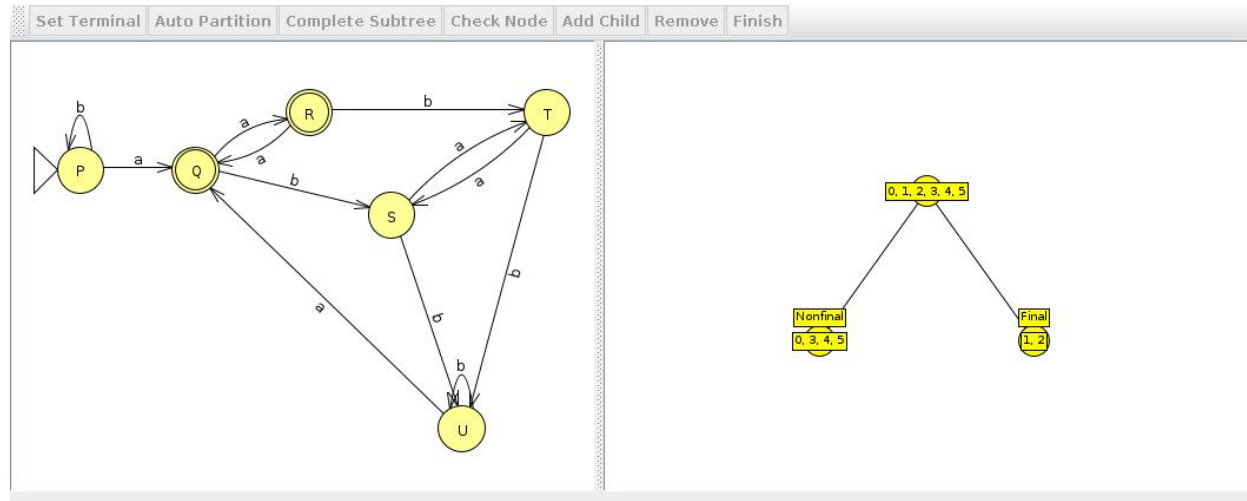
Ejercicio 3

Minimizar los siguientes autómatas finitos. Probar en JFlap

a) AFD1 = $\langle \{p, q, r, s, t, u\}, \{a, b\}, p, \delta_1, \{q, r\} \rangle$

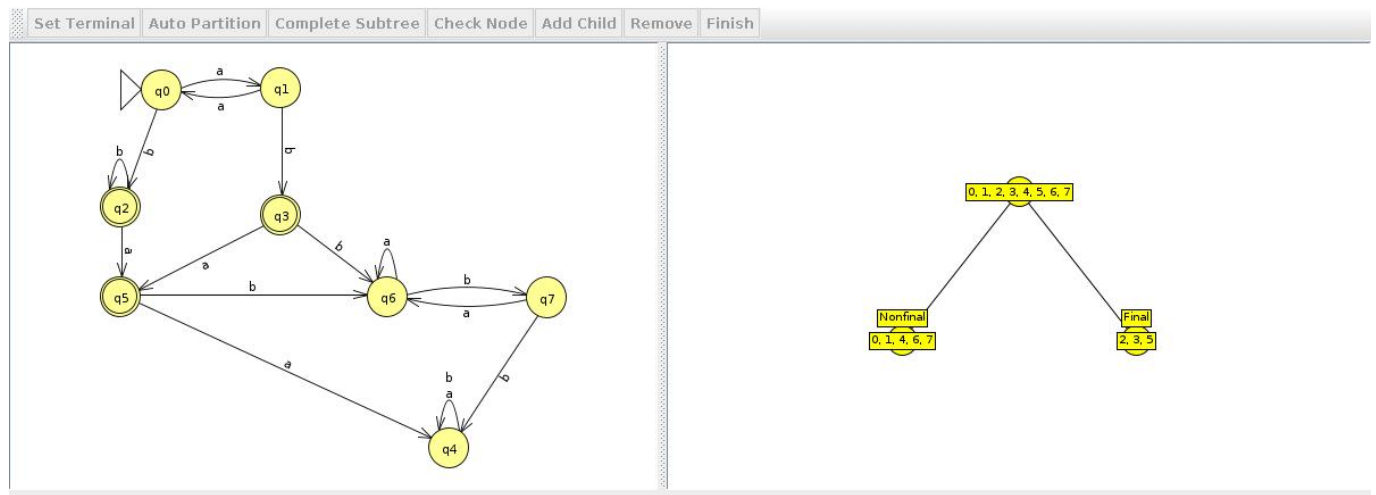
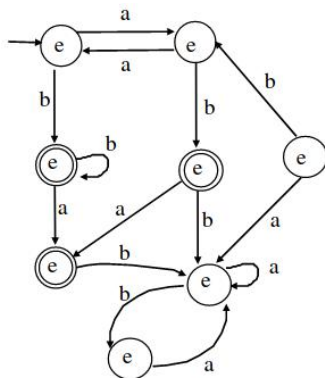
δ_1 definida por la siguiente tabla:

δ_1	a	b
P	q	p
Q	r	s
R	q	t
S	t	u
T	s	u
U	q	u



b) AFD2 = $\langle \{e0, e1, e2, e3, e4, e5, e6, e7\}, \{a, b\}, e0, \delta_2, \{e2, e3, e5\} \rangle$

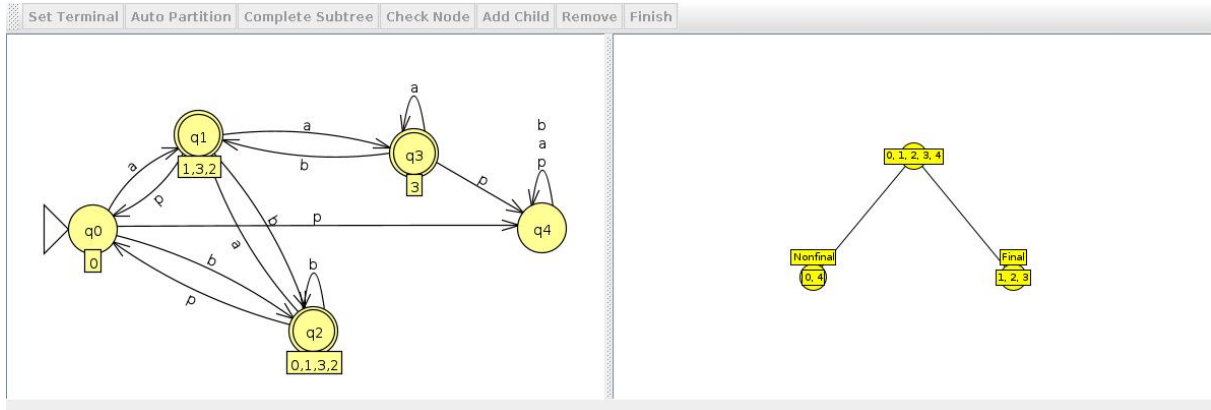
δ_2 definida por el siguiente diagrama:



c) AFND3= $\langle \{p, q, r, s\}, \{a, b\}, p, \delta_3, \{s\} \rangle$

δ_3 definida por la siguiente tabla:

δ_3	a	b
p	{q, r, s}	{p, q, r, s}
q	-	{p, q, r, s}
r	-	{p, q, r, s}
s	s	{q, r, s}



d) AFND4= $\langle \{q_0, q_1, q_2, q_3, q_4, q_5\}, \{a, b, c\}, q_0, \delta_4, \{q_2, q_5\} \rangle$

δ_4 definida como:

$$\delta_4(q_0, a) = \{q_0, q_3\}$$

$$\delta_4(q_0, b) = \{q_2\}$$

$$\delta_4(q_0, c) = \{q_5\}$$

$$\delta_4(q_1, a) = \{q_3\}$$

$$\delta_4(q_1, b) = \{q_2, q_5\}$$

$$\delta_4(q_1, c) = \{q_2\}$$

$$\delta_4(q_2, a) = \{q_2\}$$

$$\delta_4(q_2, b) = \{q_1, q_4\}$$

$$\delta_4(q_2, c) = \{q_4\}$$

$$\delta_4(q_3, a) = \{q_0\}$$

$$\delta_4(q_3, b) = \{q_5\}$$

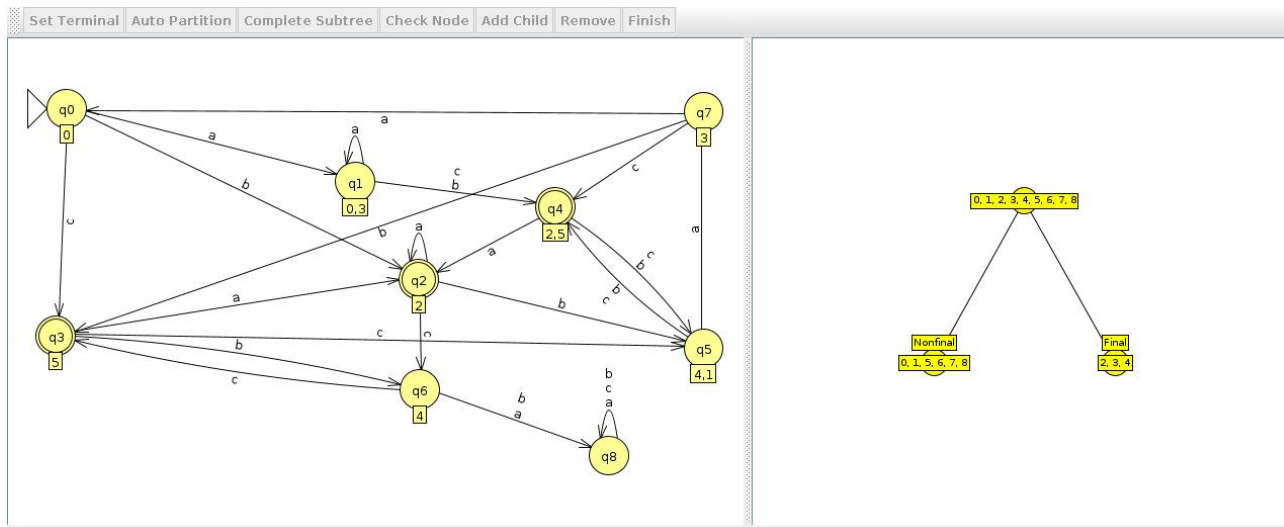
$$\delta_4(q_3, c) = \{q_2, q_5\}$$

$$\delta_4(q_4, c) = \{q_5\}$$

$$\delta_4(q_5, a) = \{q_2\}$$

$$\delta_4(q_5, b) = \{q_4\}$$

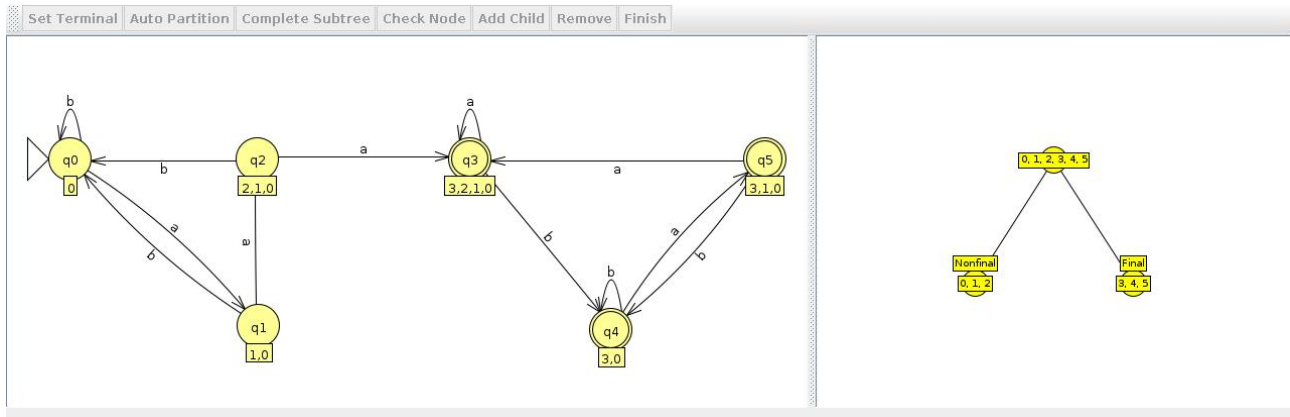
$$\delta_4(q_5, c) = \{q_1, q_4\}$$



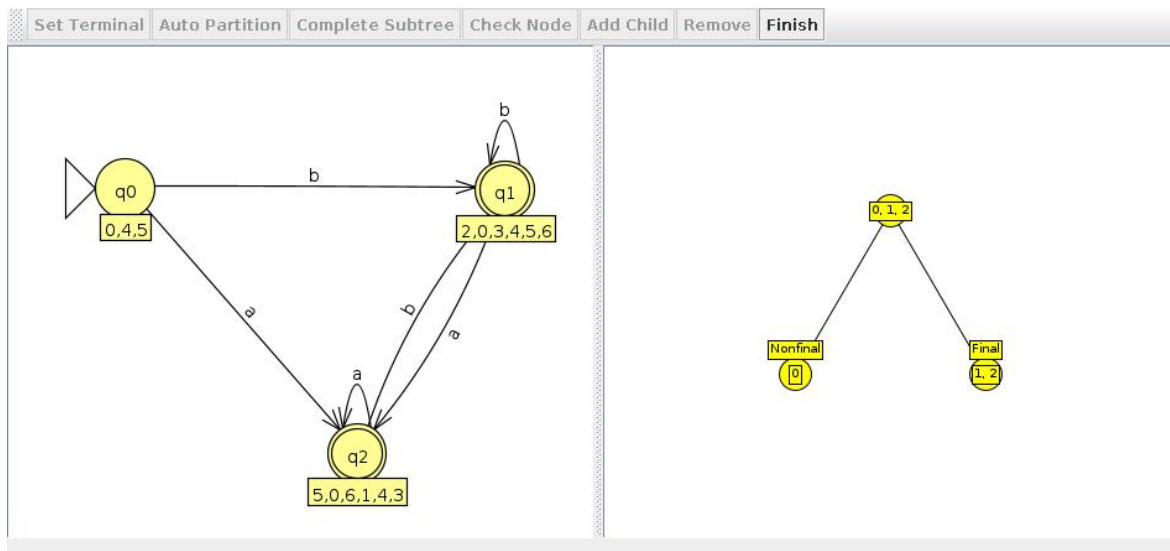
Ejercicio 4

Obtener utilizando JFlap el autómata determinístico y el autómata de estados mínimos para los siguientes autómatas:

a) $A_1 = (Q = \{q_0, q_1, q_2, q_3, \}, \Sigma = \{a, b\}, F = \{q_3\}, \delta_1)$



b) $A_2 = (Q = \{0, 1, 2, 3, 4, 5, 6\}, \Sigma = \{a, b\}, F = \{6\}, \delta_2)$



Ejercicio 5

Con el alfabeto $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ encontrar un autómata determinístico que genere números múltiplos de tres de cualquier cantidad de cifras.

Tener en cuenta que podemos subdividir el conjunto Σ en tres subconjuntos:

$S1 = \{0, 3, 6, 9\}$, $S2 = \{2, 5, 8\}$, $S3 = \{1, 4, 7\}$.

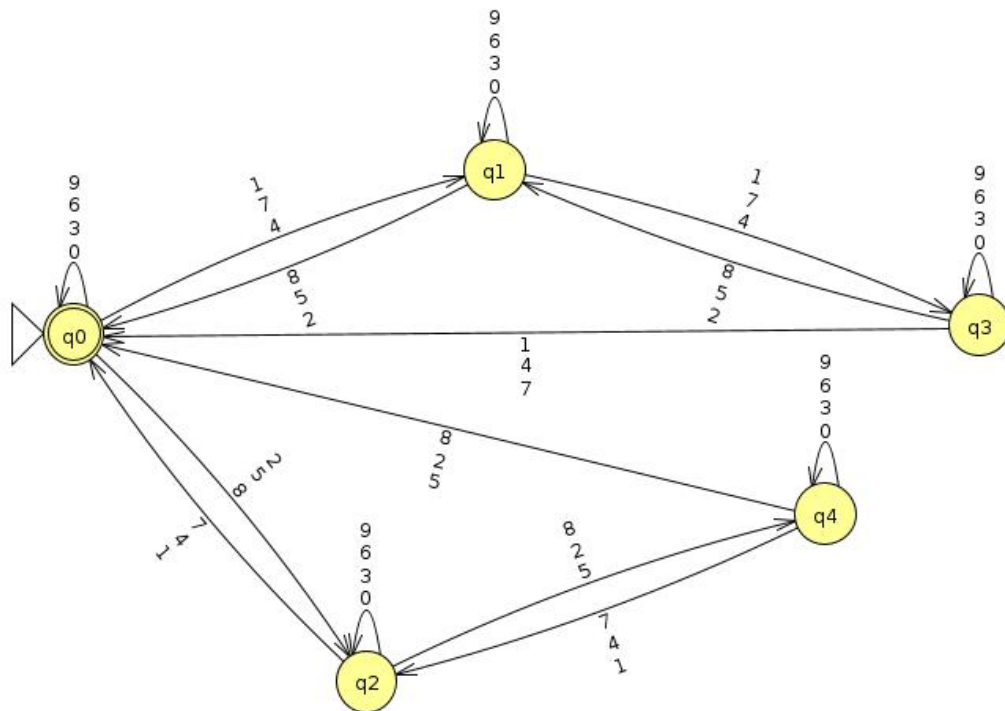
Entonces:

- Los números que se forman con la combinación de los dígitos de $S1$ son múltiplos de 3 (369, 66, 960).
- Los números que se forman con la combinación de los dígitos de $S2$ y $S3$ en igual proporción son múltiplos de 3 (1125, 4287).
- Los números que se forman con la combinación de los dígitos de $S2$ y $S3$ en igual proporción, y con cualquier número de dígitos de $S1$ son múltiplos de 3 (3021, 21567).

0,3,6,9 0369 99 36

2,5,8 y 1,4,7 \Rightarrow 2,1; 2,2,1,1; 2,4; 5,1; 2,5,1,4; 1,1,2,5; 4,2,8,7; 2,2,2;
1,4,1

2,1,3,6,9;2,4,6,6;



Ejercicio 6

Construí un autómata finito determinístico que traduzca cada cadena del lenguaje $L = \{ (ab)^n c (ba)^{2m+1} / n \geq 1, m \geq 0 \}$ en la cadena $d^{2n} . eee . (abc)^m$.

$(ab)^n c (ba)^{2m+1}$

$d^{2n} . eee . (abc)^m$

ab+

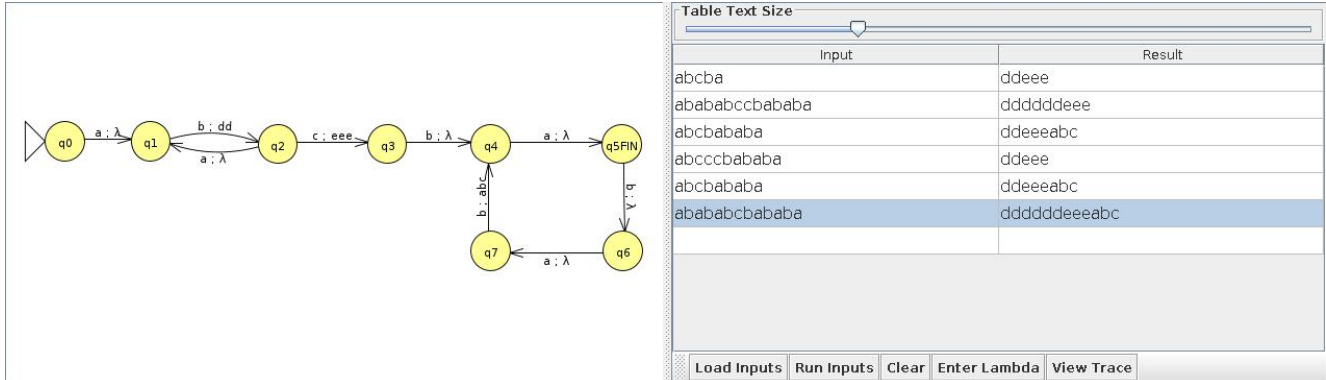
dd+

c

eee

ba(baba)*

abc*



Ejercicio 7

Se da como entrada un texto que contiene solamente letras minúsculas y los caracteres especiales \$ y _.

Diseñá un AFDT que devuelva el texto con el siguiente formato:

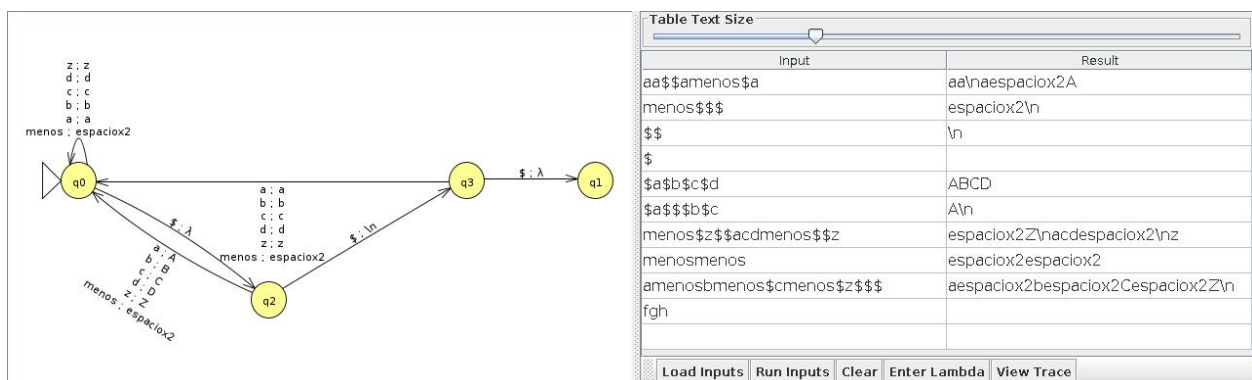
- La primera letra después de un \$, se convierte a mayúscula.
- Dos ocurrencias consecutivas de \$ se transforman en un salto de línea.
- El caracter _ se reemplaza por dos espacios en blanco.
- En el texto de entrada no pueden darse más de dos ocurrencias consecutivas del \$, excepto una secuencia de tres \$ que indica el fin de la cadena.
- El signo \$ y el _ no deben aparecer en el texto de salida.

$\Sigma = \{a, b, c, \dots, z, \$, _ \}$.

hola\$como_va\$\$\$ => holaComo va.

\$hola_\$como_\$va\$\$\$ => Hola Como Va.

\$hola_\$como_\$va\$\$bien_\$vos\$\$bien_\$chau\$\$\$ => Hola Como Va
bien Vos
bien Chau.

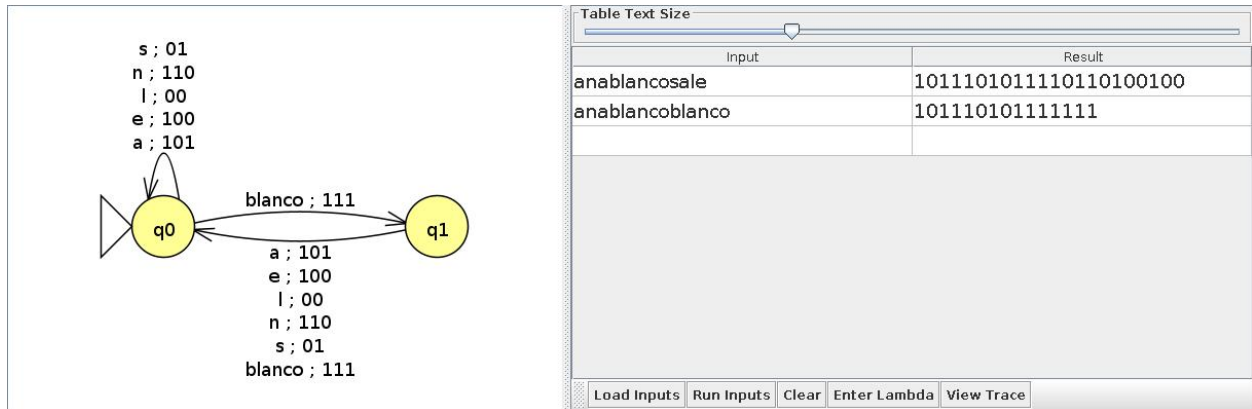


Ejercicio 8

Dada la siguiente codificación de caracteres:

- a) blanco = 111.
- b) a = 101.
- c) e = 100.
- d) l = 00.
- e) n = 110.
- f) s = 01.

Por ejemplo, el mensaje *ana sale* se codifica como 1011101011110110100100. Construí un autómata finito que dado un mensaje codificado lo devuelva decodificado.



Ejercicio 9

Se desea modelar el comportamiento de una máquina expendedora de boletos de colectivo.

- El precio de cada boleto es \$1.
- La máquina acepta monedas de \$0.25 y \$0.50; y devuelve el cambio necesario.
- Para comprar un boleto se deben introducir las monedas, y luego apretar el botón B para solicitarlo.

