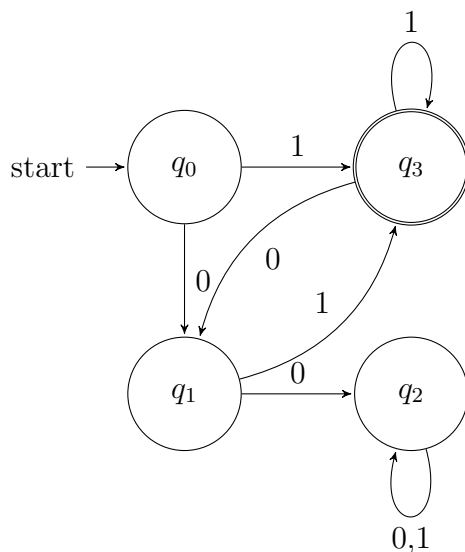


Problema 1. Definir autómatas deterministas con cuatro estados que reconozcan los siguientes lenguajes.

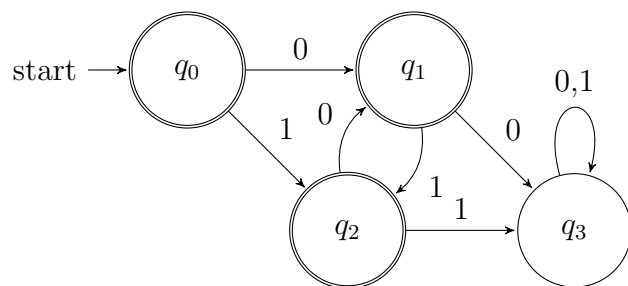
- (a)  $\{x \in \{0,1\}^* : x \text{ acaba en } 1 \text{ y no contiene } 00\}$ .
- (b)  $\{x \in \{0,1\}^* : x \text{ no contiene ni } 00 \text{ ni } 11\}$ .
- (c)  $\{x \in \{0,1\}^* : \text{cada } 0 \text{ en } x \text{ va inmediatamente precedido e inmediatamente seguido por un } 1\}$ .

**Solución:**

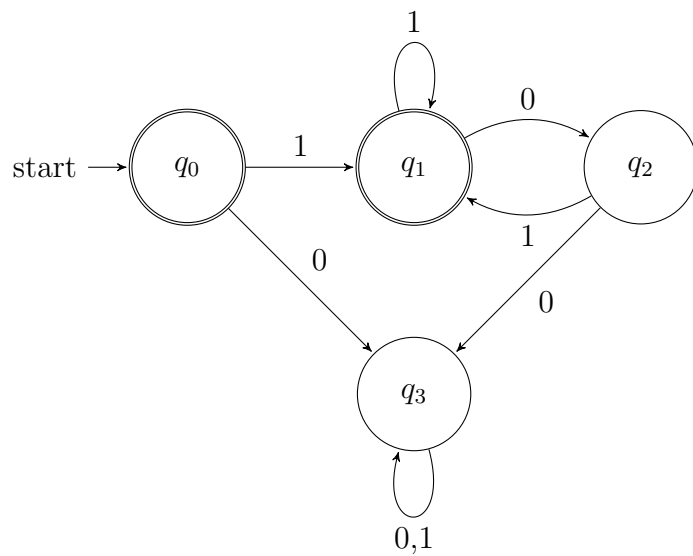
(a) Definimos el siguiente autómata determinista, que consta de los estados  $q_0$ ,  $q_1$ ,  $q_2$  y  $q_3$ , donde  $q_0$  es el estado inicial y  $q_3$  es el único estado aceptador.



(b) Definimos el siguiente autómata determinista, que consta de los estados  $q_0$ ,  $q_1$ ,  $q_2$  y  $q_3$ , donde  $q_0$  es el estado inicial y  $q_0$ ,  $q_1$  y  $q_2$  son los estados aceptadores.



(c) Definimos el siguiente autómata determinista, que consta de los estados  $q_0$ ,  $q_1$ ,  $q_2$  y  $q_3$ , donde  $q_0$  es el estado inicial y  $q_0$  y  $q_1$  son los estados aceptadores.

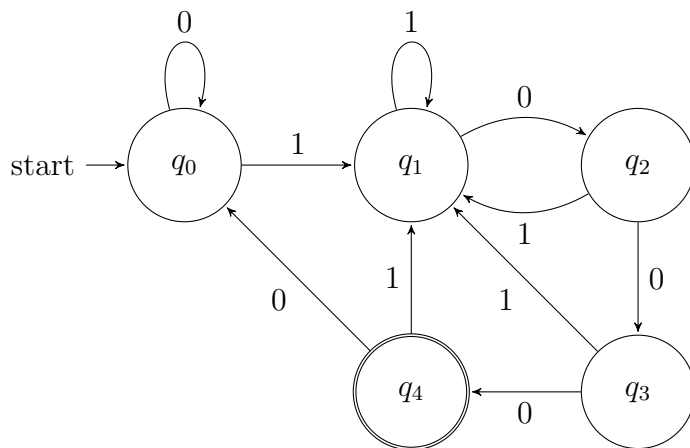


Problema 2. (a) Definir un autómata determinista que reconozca el lenguaje de las palabras de bits que acaban en 1000.

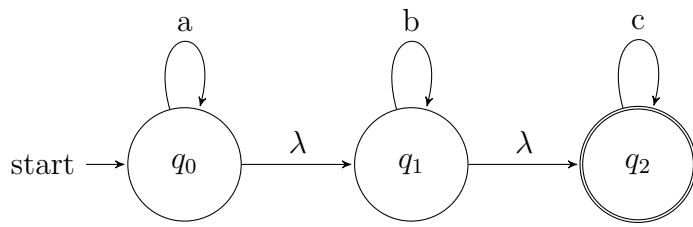
(b) Definir un autómata indeterminista que reconozca el lenguaje de las palabras con cero o más letras a, seguidas de cero o más letras b, seguidas de cero o más letras c .

**Solución:**

(a) Definimos el autómata por el siguiente gráfico, donde  $q_4$  es el único estado aceptador.



(b) Definimos el siguiente autómata indeterminista, que consta de los estados  $q_0$ ,  $q_1$  y  $q_2$ , donde  $q_0$  es el estado inicial y  $q_2$  es el único estado aceptador.

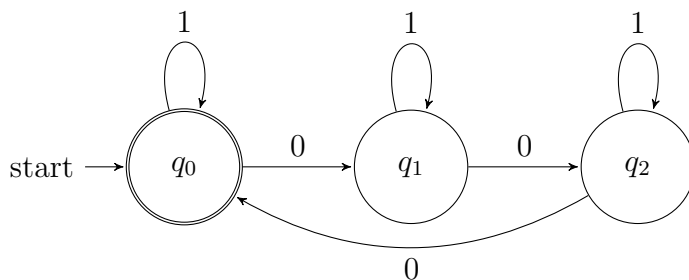


Problema 3. (a) Definir un autómata determinista  $M$  tal que  $L(M) = \{x \in \{0, 1\}^* : n_0(x) \text{ es un múltiplo de } 3\}$ .

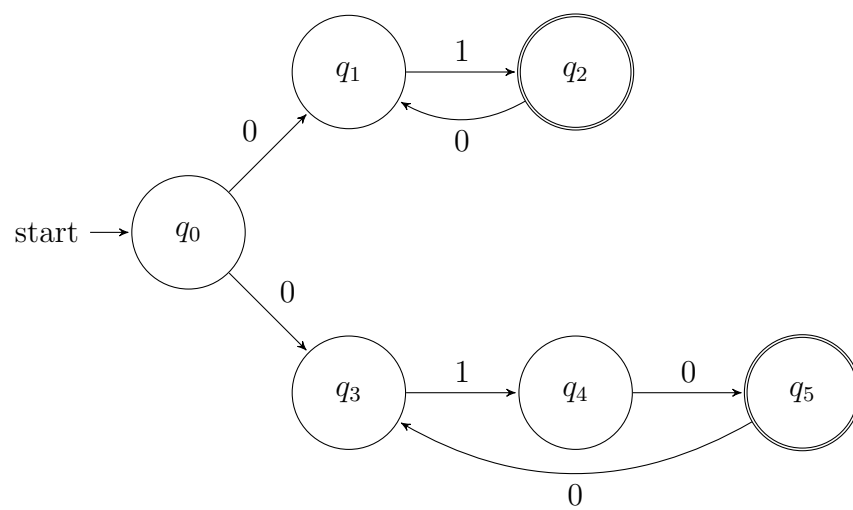
(b) Definir un autómata indeterminista  $M$  tal que  $L(M) = \{(01)^n : n \geq 1\} \cup \{(010)^n : n \geq 1\}$ , es decir,  $L(M)$  es el lenguaje de las palabras formadas por 01 repetido una o más veces, o por 010 repetido una o más veces.

**Solución:**

(a) Definimos el siguiente autómata determinista, que consta de los estados  $q_0$ ,  $q_1$  y  $q_2$ , donde  $q_0$  es el estado inicial y es asimismo el único estado aceptador.



(b) Definimos el siguiente autómata indeterminista, que consta de los estados  $q_0$ ,  $q_1$ ,  $q_2$ ,  $q_3$ ,  $q_4$  y  $q_5$ , donde  $q_0$  es el estado inicial y  $q_2$  y  $q_5$  son los estados aceptadores.

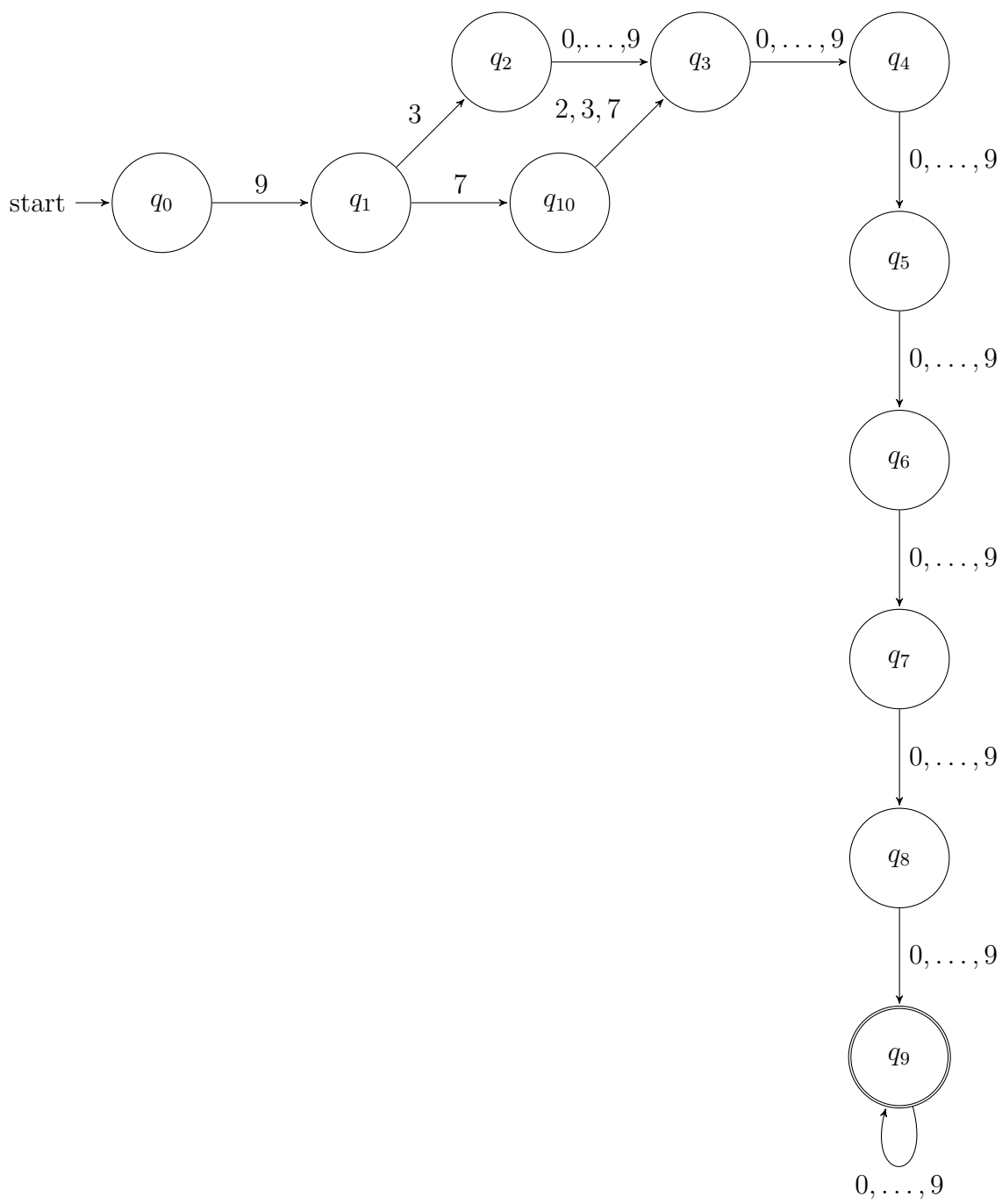


Problema 4. (a) Explicar cómo se puede construir un autómata indeterminista para reconocer los números de teléfono de las provincias de Catalunya.

(b) Explicar cómo a partir del l'autómata del apartado (a), se puede escribir un programa en JAVA para reconocer tales números

**Solución:**

(a) Recordemos que el prefijo para llamar a Girona es el 972, a Lleida es el 973 y a Tarragona es el 977. El siguiente autómata indeterminista, en donde  $q_9$  es el único estado aceptador, reconoce los números de teléfono de Catalunya.





(b) Para escribir un programa en JAVA que reconozca los números de teléfono de Catalunya, en primer lugar hemos de eliminar el indeterminismo del autómata del apartado (a). Podemos hacer esto directamente añadiendo un estado de error al autómata, al cual llegaremos desde cualquier otro estado cuando entre un dígito distinto a los indicados en las transiciones del gráfico del autómata del apartado (a). El programa asociado a este autómata determinista será entonces el programa buscado para reconocer los números de teléfono de Catalunya.

**Problema 5.** Consideremos el autómata indeterminista  $M = (\{q_0, q_1, q_2, q_3, q_4, q_5\}, \{\text{digito}\}, \Delta, q_0, \{q_5\})$  donde  $\Delta$  está definida por la siguiente tabla:

$q_0$	digito	$q_1$
$q_1$	$\lambda$	$q_2$
$q_2$	$\lambda$	$q_3$
$q_2$	$\lambda$	$q_5$
$q_3$	digito	$q_4$
$q_4$	$\lambda$	$q_3$
$q_4$	$\lambda$	$q_5$

Se pide entonces:

- (1) Calcular los  $\lambda$ -cierres de los estados de  $M$ .
- (2) Utilizando el algoritmo visto en clase, transformar el autómata  $M$  en un autómata determinista equivalente.

**Solución:**

(1) Recordad que para todo estado  $q$ , el  $\lambda$ -cierre de  $q$  es el conjunto de todos los estados  $p$  para los cuales hay un cómputo en el autómata desde el estado  $q$  hasta el estado  $p$  sin leer ningún símbolo de la entrada. Tenemos entonces :

$$\Lambda(q_0) = \{q_0\},$$

$$\Lambda(q_1) = \{q_1, q_2, q_3, q_5\},$$

$$\Lambda(q_2) = \{q_2, q_3, q_5\},$$

$$\Lambda(q_3) = \{q_3\},$$

$$\Lambda(q_4) = \{q_3, q_4, q_5\},$$

$$\Lambda(q_5) = \{q_5\}.$$

(2) El estado inicial del autómata determinista  $M'$  equivalente a  $M$  es  $\Lambda(q_0) = \{q_0\}$ . Calculamos ahora la función de transición  $\delta'$  de  $M'$ . Obsérvese que el único símbolo terminal es digito. Tenemos entonces:

$$\delta'(\{q_0\}, \text{digito}) = \Lambda(q_1) = \{q_1, q_2, q_3, q_5\},$$

$$\delta'(\{q_1, q_2, q_3, q_5\}, \text{digito}) = \Lambda(q_4) = \{q_3, q_4, q_5\},$$

$$\delta'(\{q_3, q_4, q_5\}, \text{digito}) = \Lambda(q_4) = \{q_3, q_4, q_5\}.$$

Como  $q_5$  es el único estado aceptador del autómata  $M$ , los estados aceptadores de  $M'$  son los estados que contienen a  $q_5$ , es decir,  $\{q_1, q_2, q_3, q_5\}$  y  $\{q_3, q_4, q_5\}$ . Para simplificar la notación, pongamos  $p_0 = \{q_0\}$ ,  $p_1 = \{q_1, q_2, q_3, q_5\}$  y  $p_2 = \{q_3, q_4, q_5\}$ . Se observa ahora que los estados  $p_1$  y  $p_2$  son equivalentes, ya que desde cualquiera de los dos estados  $p_1$ ,  $p_2$ , nunca volvemos al único estado no aceptador  $p_0$ , y por tanto el lenguaje del autómata simplificado con un único estado aceptador es el mismo que el lenguaje del autómata con los dos estados aceptadores  $p_1$  y  $p_2$ . Por tanto, obtenemos el autómata determinista formado por dos estados  $p_0$  y  $p_1$  donde  $p_0$  es el estado inicial y  $p_1$  es el único estado aceptador y en donde la función de transición está definida por  $\delta'(p_0, \text{digito}) = p_1$  y  $\delta'(p_1, \text{digito}) = p_1$ . Por tanto, el autómata reconoce los números enteros sin signo.

Problema 6. Consideremos el autómata indeterminista  $M = (\{P, Q, R, S\}, \{a, b\}, \Delta, P, \{P, Q\})$  donde  $\Delta$  está definida por la siguiente tabla:

$P$	$a$	$S$
$P$	$a$	$Q$
$P$	$\lambda$	$Q$
$Q$	$b$	$Q$
$Q$	$\lambda$	$R$
$R$	$b$	$P$
$S$	$a$	$S$
$S$	$b$	$R$

Siguiendo el método visto en clase, transformar el autómata  $M$  en un autómata determinista equivalente.

**Solución:**

Calculamos primero los cierres de los estados. Se tiene que  $\Lambda(P) = PQR$ ,  $\Lambda(Q) = QR$ ,  $\Lambda(R) = R$  y  $\Lambda(S) = S$ . Construimos entonces el autómata determinista  $M'$  equivalente a  $M$ . El estado inicial de  $M'$  es  $\Lambda(P) = PQR$ . Construimos la función de transición  $\delta'$  para  $M'$ .

$$\begin{aligned}
\delta'(PQR, a) &= \Lambda(Q) \cup \Lambda(S) = QRS, \\
\delta'(PQR, b) &= \Lambda(P) \cup \Lambda(Q) = PQR, \\
\delta'(QRS, a) &= \Lambda(S) = S, \\
\delta'(QRS, b) &= \Lambda(P) \cup \Lambda(Q) \cup \Lambda(R) = PQR, \\
\delta'(S, a) &= \Lambda(S) = S, \\
\delta'(S, b) &= \Lambda(R) = R, \\
\delta'(R, a) &= \emptyset, \\
\delta'(R, b) &= \Lambda(P) = PQR, \\
\delta'(\emptyset, 0) &= \delta'(\emptyset, 1) = \emptyset.
\end{aligned}$$

Por tanto, los estados de  $M'$  son:  $PQR, QRS, S, R$  y  $\emptyset$ . Como  $P$  y  $Q$  son los estados aceptadores de  $M$ , los estados aceptadores de  $M'$  son  $PQR$  y  $QRS$ .

Problema 7. Consideremos el autómata indeterminista  $M = (\{A, B, C, D, E\}, \{0, 1\}, \Delta, A, \{B, C\})$  donde  $\Delta$  está definida por la siguiente tabla:

$A$	$\lambda$	$B$
$A$	$\lambda$	$C$
$B$	$0$	$D$
$B$	$1$	$B$
$C$	$0$	$C$
$C$	$1$	$E$
$D$	$0$	$B$
$D$	$1$	$D$
$E$	$0$	$E$
$E$	$1$	$C$

Se pide entonces:

- (1) Describir el lenguaje  $L(M)$ .
- (2) Siguiendo el método visto en clase, transformar el autómata  $M$  en un autómata determinista equivalente.
- (3) Programar en JAVA el autómata determinista obtenido en (2).

**Solución:**

(1) Se observa que el estado  $B$  reconoce las palabras de bits que tienen un número par de ceros, y el estado  $C$  reconoce las palabras de bits que tienen un número par de unos. Por tanto, como  $B$  y  $C$  son los estados aceptadores de  $M$ , tenemos que

$$L(M) = \{x \in \{0, 1\}^* : n_0(x) \text{ es par o } n_1(x) \text{ es par}\}.$$

(2) Se tiene que  $\Lambda(A) = \{A, B, C\} = ABC$ ,  $\Lambda(B) = B$ ,  $\Lambda(C) = C$ ,  $\Lambda(D) = D$  y  $\Lambda(E) = E$ . Construimos entonces el autómata determinista  $M'$  equivalente a  $M$ . El estado inicial de  $M'$  es  $\Lambda(A) = ABC$ . Definimos ahora la función de transición  $\delta'$  para  $M'$ .

$$\begin{aligned}\delta'(ABC, 0) &= \Lambda(C) \cup \Lambda(D) = CD, \\ \delta'(ABC, 1) &= \Lambda(B) \cup \Lambda(E) = BE, \\ \delta'(CD, 0) &= \Lambda(B) \cup \Lambda(C) = BC,\end{aligned}$$

$$\begin{aligned}
\delta'(CD, 1) &= \Lambda(D) \cup \Lambda(E) = DE, \\
\delta'(BE, 0) &= \Lambda(D) \cup \Lambda(E) = DE, \\
\delta'(BE, 1) &= \Lambda(B) \cup \Lambda(C) = BC, \\
\delta'(BC, 0) &= \Lambda(C) \cup \Lambda(D) = CD, \\
\delta'(BC, 1) &= \Lambda(B) \cup \Lambda(E) = BE, \\
\delta'(DE, 0) &= \Lambda(B) \cup \Lambda(E) = BE, \\
\delta'(DE, 1) &= \Lambda(C) \cup \Lambda(D) = CD.
\end{aligned}$$

Por tanto, los estados de  $M'$  son:  $ABC, CD, BE, BC$  y  $DE$ . Como  $B$  y  $C$  son los estados aceptadores de  $M$ , los estados aceptadores de  $M'$  son  $ABC, CD, BE$  y  $BC$ .

(3) Representamos a  $ABC$  por 0, a  $CD$  por 1, a  $BE$  por 2, a  $BC$  por 3 y a  $DE$  por 4. Podemos escribir entonces el siguiente programa en JAVA para simular el autómata  $M'$ :

```

public boolean simular (String entrada)
{ int q = 0, i = 0;
  char c = entrada.charAt(0);
  while (c != '$')
  { switch(q)
    { case 0:
      if (c == '0') q = 1; else if (c == '1') q = 2;
      break;
      case 1:
      if (c == '0') q = 3; else if (c == '1') q = 4;
      break;
      case 2:
      if (c == '0') q = 4; else if (c == '1') q = 3;
      break;
      case 3:
      if (c == '0') q = 1; else if (c == '1') q = 2;
      break;
      case 4:
      if (c == '0') q = 2; else if (c == '1') q = 1;
      break; }
    c = entrada.charAt(++i); }
  if (q == 4) return false; else return true; }

```

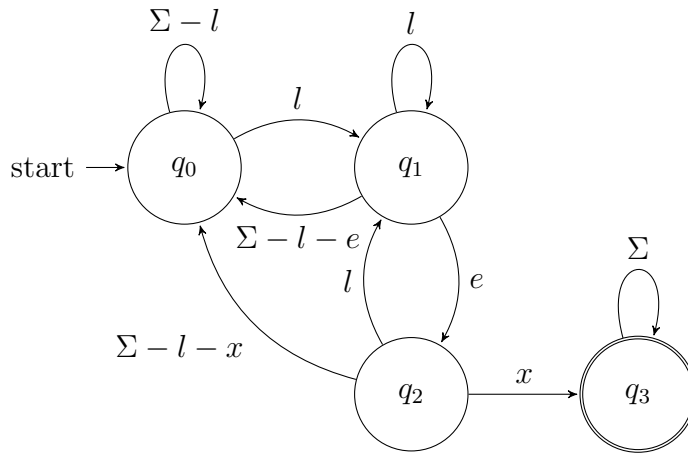
Problema 8. (a) Definir un autómata determinista que determine si un texto de caracteres contiene el patrón "lex".

(b) Definir un autómata indeterminista que determine si un texto de caracteres contiene el patrón "lex" o el patrón "ens2001".

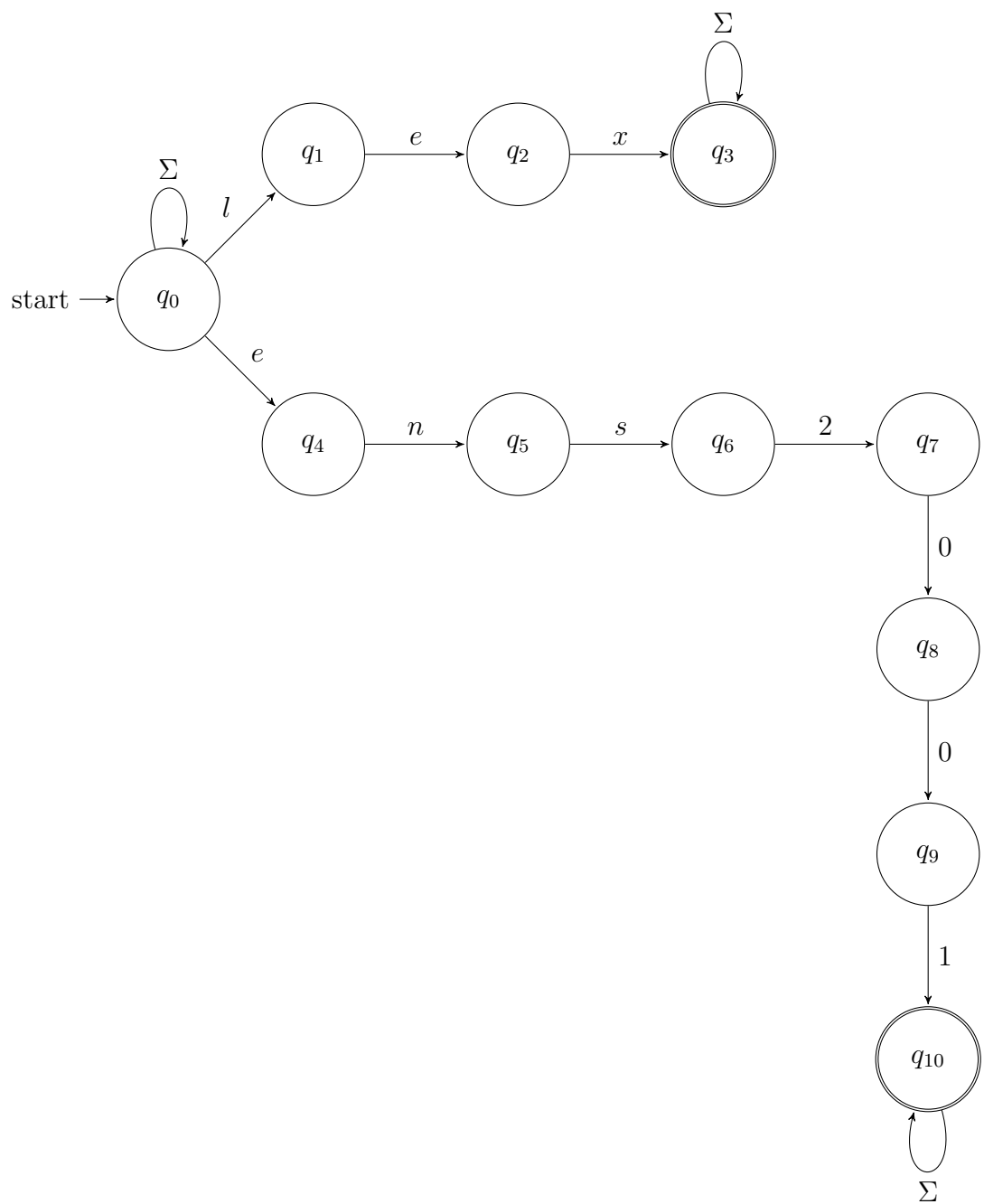
(c) Programar en Java el autómata determinista definido en (a).

**Solución:**

(a) Denotamos por  $\Sigma$  al conjunto de los caracteres ASCII. El siguiente autómata determinista determina entonces si un texto de caracteres contiene el patrón "lex".



(b) Denotamos por  $\Sigma$  al conjunto de los caracteres ASCII. El siguiente autómata indeterminista determina entonces si un texto de caracteres contiene el patrón "lex" o el patrón "ens2001".





(c) Podemos escribir el siguiente programa en JAVA para simular el autómata determinista del apartado (a).

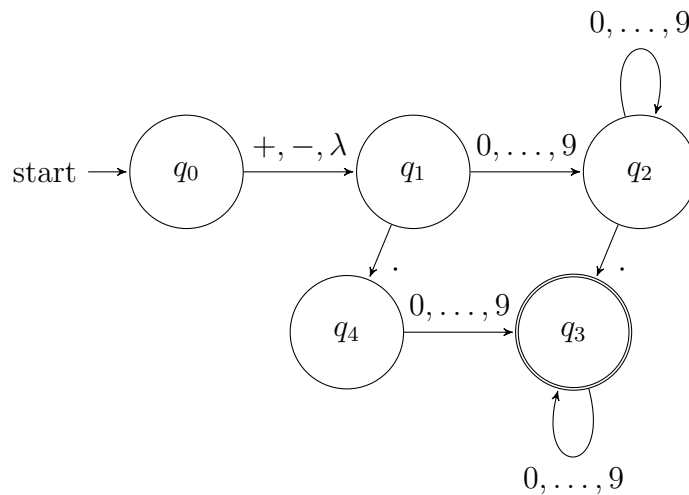
```
public boolean simular (String entrada)
{ int q = 0, i = 0;
  char c = entrada.charAt(0);
  while (c != '$')
  { switch(q)
    { case 0:
      if (c == 'l') q = 1;
      break;
      case 1:
      if (c == 'e') q = 2; else if ((c != 'e') && (c != 'l')) q = 0;
      break;
      case 2:
      if (c == 'x') return true; else if (c == 'l') q = 1; else q = 0;
      break; }
    c = entrada.charAt(++i); }
  return false; }
```

Problema 9. (a) Definir un autómata indeterminista para reconocer números decimales que contengan: (a) un signo  $+$  o  $-$  opcional; (b) una palabra de dígitos; (c) un punto decimal; (d) una segunda palabra de dígitos. Tanto la primera palabra de dígitos como la segunda pueden estar vacías, pero al menos una de las dos palabras no puede estar vacía.

(b) Explicar, cómo utilizando el autómata del apartado (a), se puede escribir un programa en JAVA para reconocer números decimales (el tipo “float”).

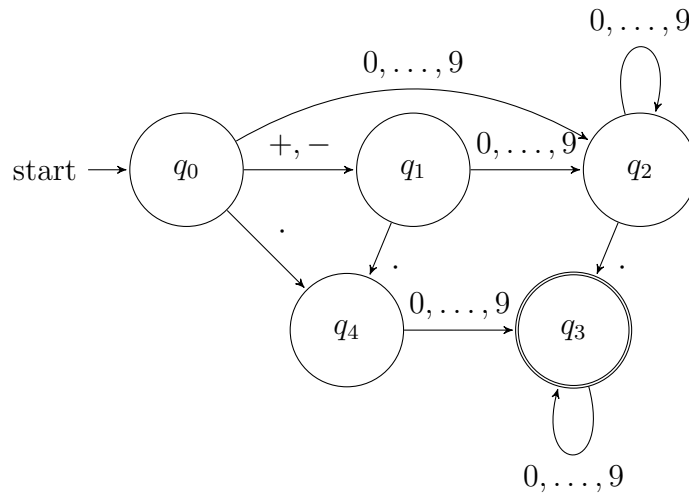
**Solución:**

(a) En primer lugar, definimos el siguiente autómata indeterminista  $M_1$ , en donde  $q_0$  es el estado inicial, y  $q_3$  es el único estado aceptador.



(b) Para escribir un programa en JAVA que reconozca los números decimales, en primer lugar hemos de eliminar el indeterminismo del autómata  $M_1$  del apartado (a). Para ello, podemos utilizar el algoritmo visto en clase de teoría para transformar un autómata indeterminista en un autómata determinista equivalente. Sin embargo, en este caso podemos construir di-

rectamente el autómata determinista equivalente. Para ello, construimos el siguiente autómata  $M_2$  equivalente a  $M_1$ , en el que eliminamos la transición con la  $\lambda$ .



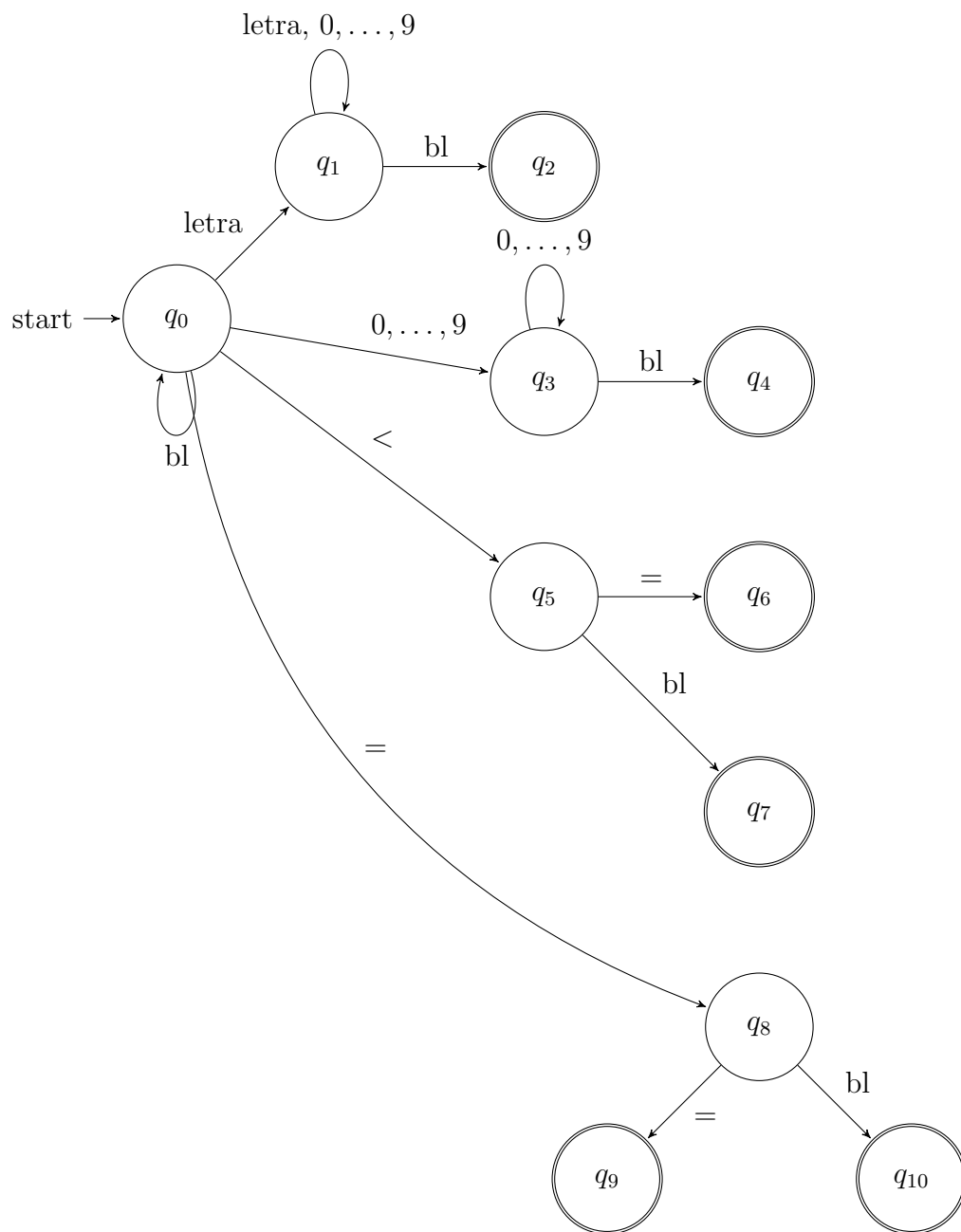
Ahora, podemos definir el autómata determinista equivalente, simplemente añadiendo a este último autómata  $M_2$  un estado de error, al cual llegaremos desde cualquier otro estado cuando entre un carácter distinto a los indicados en las transiciones del gráfico del autómata  $M_2$ . El programa asociado a este autómata determinista será entonces el programa buscado para reconocer los números decimales.

Problema 10. Explicar cómo diseñar un analizador léxico para reconocer las siguientes categorías sintácticas:

- (i) identificadores formados por letras y dígitos de manera que el primer carácter es una letra,
- (ii) números enteros sin signo,
- (iii) la asignación  $=$ ,
- (iv) los predicados  $==$ ,  $<$  y  $<=$ .

**Solución:**

En primer lugar, construimos el siguiente autómata indeterminista  $M$  para reconocer las categorías sintácticas indicadas, en el cual los estados aceptadores son  $q_2$ ,  $q_4$ ,  $q_6$ ,  $q_7$ ,  $q_9$  y  $q_{10}$ . Representamos por  $bl$  al carácter blanco.



Por tanto, el estado  $q_2$  reconoce un identificador formado por letras y dígitos, el estado  $q_4$  reconoce un entero sin signo, el estado  $q_6$  reconoce  $\leq$ , el estado  $q_7$  reconoce  $<$ , el estado  $q_9$  reconoce  $==$  y el estado  $q_{10}$  reconoce

la asignación  $=$ . Se observa que los estados  $q_2$ ,  $q_4$ ,  $q_7$  y  $q_{10}$  van adelantados un carácter, ya que dichos estados reconocen la categoría sintáctica al leer el símbolo siguiente a la categoría. Entonces, para escribir el analizador léxico, en primer lugar hemos de añadir un estado de error al autómata  $M$ , al cual llegaremos desde cualquier otro estado cuando entre un carácter distinto a los indicados en las transiciones del gráfico del autómata  $M$ . A continuación, programamos el autómata resultante, siguiendo el método visto en clase, con las siguientes modificaciones:

- Cuando se llegue a un estado aceptador, el programa imprimirá la categoría sintáctica reconocida y volverá al estado inicial.
- Cuando se llegue a  $q_2$ , a  $q_4$ , a  $q_7$  o a  $q_{10}$ , el programa no leerá el siguiente símbolo de la entrada. Y sí lo leerá, cuando llegue a cualquier otro estado.