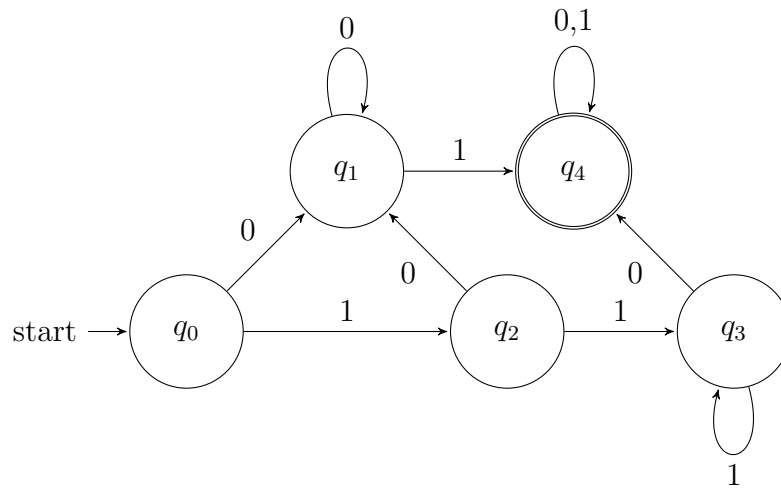
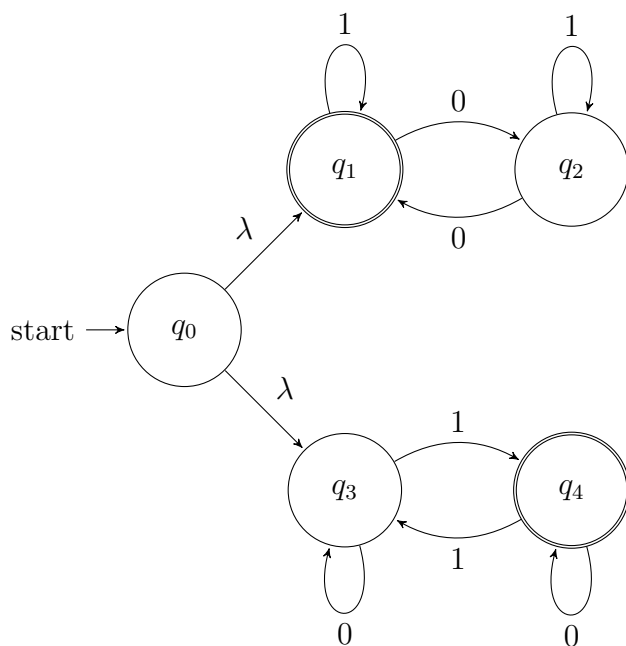


Problema 1. (a) Describir el lenguaje reconocido por el siguiente autómata determinista  $M$ , donde  $q_4$  es el único estado aceptador.



(b) Describir el lenguaje reconocido por el siguiente autómata indeterminista, donde  $q_1$  y  $q_4$  son los estados aceptadores.



**Solución:**

(a)  $L(M) = \{x \in \{0, 1\}^* : x \text{ contiene } 01 \text{ o } 110 \text{ como subpalabra}\}$ . También,  $L(M)$  se puede representar por la expresión regular  $\alpha = (0 \cup 1)^*(01 \cup 110)(0 \cup 1)^*$ .

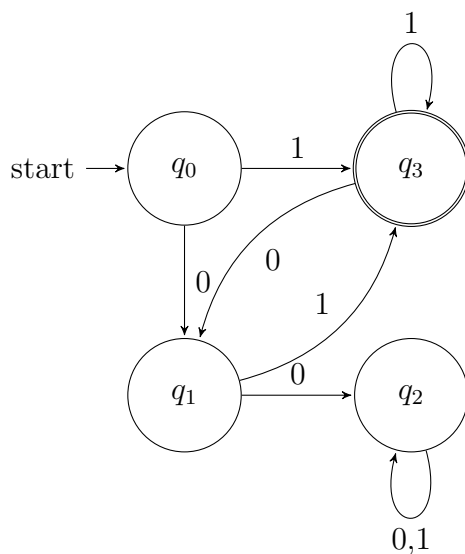
(b) El estado  $q_1$  reconoce el lenguaje de las palabras de bits en las cuales el número de ceros es par. Y el estado  $q_4$  reconoce el lenguaje de las palabras de bits en las cuales el número de unos es impar. Por tanto, como  $q_1$  y  $q_4$  son los estados aceptadores, se tiene que el lenguaje reconocido por el autómata es  $L = \{x \in \{0, 1\}^* : n_0(x) \text{ es par } \text{ ó } n_1(x) \text{ es impar}\}$ .

Problema 2. Definir autómatas deterministas que reconozcan los siguientes lenguajes.

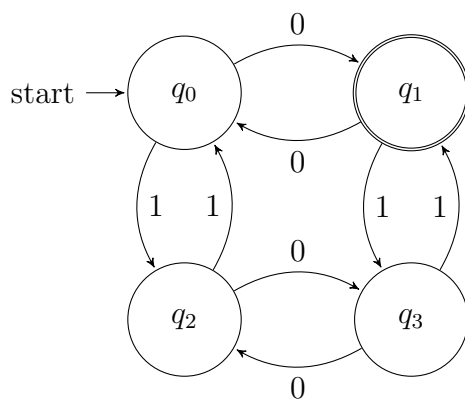
- (a)  $\{x \in \{0,1\}^* : x \text{ acaba en } 1 \text{ y no contiene } 00\}$ .
- (b)  $\{x \in \{0,1\}^* : x \text{ tiene un número impar de ceros y un número par de unos }\}$ .

**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  $q_1$  es el único estado aceptador.



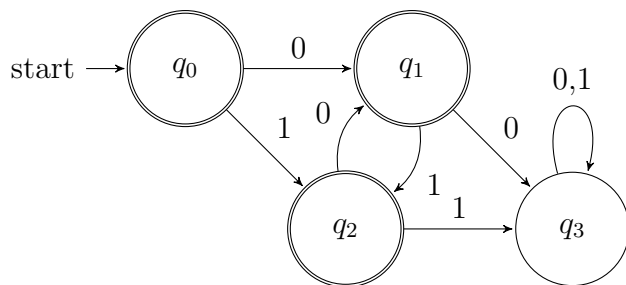
Obsérvese que estaremos en el estado  $q_0$ , cuando el número de ceros leídos sea par y el número de unos leídos sea par; estaremos en el estado  $q_1$ , cuando el número de ceros leídos sea impar y el número de unos leídos sea par; estaremos en el estado  $q_2$ , cuando el número de ceros leídos sea par y el número de unos leídos sea impar; y estaremos en el estado  $q_3$ , cuando el número de ceros leídos sea impar y el número de unos leídos sea impar. Como  $q_1$  es el único estado aceptador del autómata, se tiene que el lenguaje del autómata es el conjunto  $\{x \in \{0,1\}^* : x \text{ tiene un número impar de ceros y un número par de unos}\}$ .

Problema 3. (a) Definir un autómata determinista  $M$  tal que  $L(M) = \{x \in \{0, 1\}^* : x \text{ no contiene ni } 00 \text{ ni } 11\}$ .

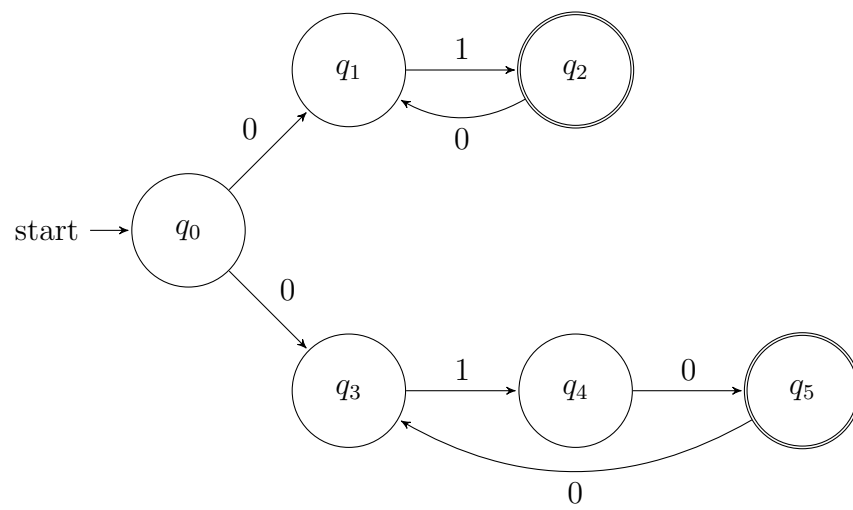
(b) Definir un autómata indeterminista  $M$  tal que  $L(M) = \{(01)^n : n \geq 1\} \cup \{(010)^n : n \geq 1\}$ , es decir,  $L(M)$  es el lenguaje de las palabras formadas por 01 repetido una o más veces, o por 010 repetido una o más veces.

**Solución:**

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



(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.



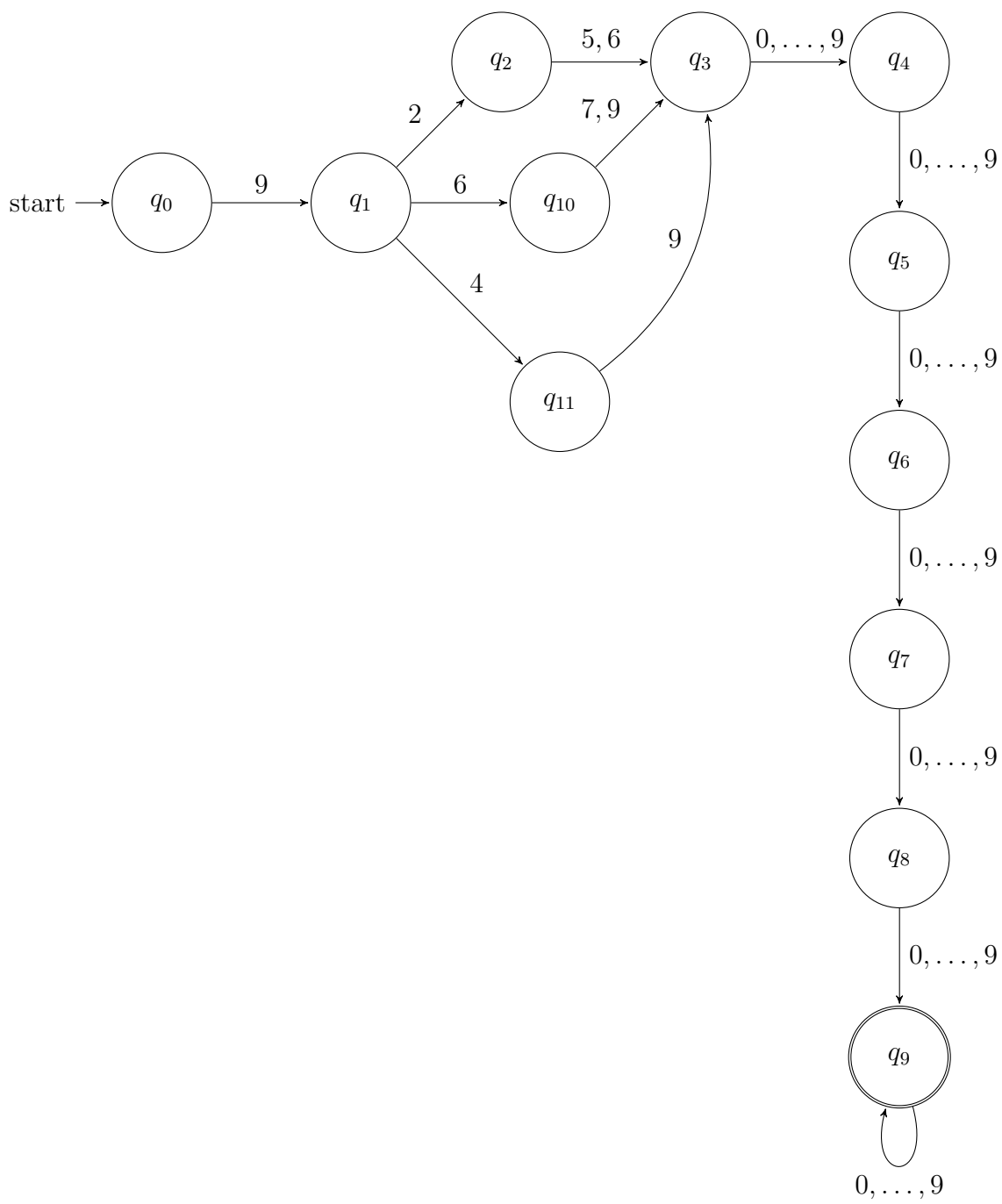
Problema 4. Para llamar por teléfono a una provincia de la comunidad de Castilla-La Mancha hay que marcar un número de seis dígitos y previamente el siguiente prefijo: 967 si se llama a Albacete, 969 si se llama a Cuenca, 949 si se llama a Guadalajara, 925 si se llama a Toledo, y 926 si se llama a Ciudad Real. Se pide entonces:

(a) Definir un autómata indeterminista que reconozca los números de teléfono de la comunidad de Castilla-La Mancha.

(b) Explicar cómo, utilizando el autómata del apartado (a), se puede escribir un programa en JAVA para reconocer los números de teléfono de Castilla-La Mancha.

**Solución:**

(a) El siguiente autómata indeterminista, en donde  $q_9$  es el único estado aceptador, reconoce los números de teléfono de Castilla-La Mancha.





(b) Para escribir un programa en JAVA que reconozca los números de teléfono de Castilla-La Mancha, 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 Castilla-La Mancha.

Problema 5. 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 6. Consideremos el autómata indeterminista  $M = (\{A, B, C, D\}, \{0, 1\}, \Delta, A, \{D\})$  donde  $\Delta$  está definida por la siguiente tabla:

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

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 o en C el autómata determinista obtenido en (2).

**Solución:**

(1)  $L(M) = \{0^m(01)^n0^k : m, n, k \geq 0\}$ . También,  $L(M)$  se puede representar por la expresión regular  $0^*(01)^*0^*$ .

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

$$\begin{aligned}
 \delta'(ABD, 0) &= \Lambda(A) \cup \Lambda(C) \cup \Lambda(D) = ABCD, \\
 \delta'(ABD, 1) &= \emptyset, \\
 \delta'(ABCD, 0) &= \Lambda(A) \cup \Lambda(C) \cup \Lambda(D) = ABCD, \\
 \delta'(ABCD, 1) &= \Lambda(B) = BD, \\
 \delta'(\emptyset, 0) &= \delta'(\emptyset, 1) = \emptyset, \\
 \delta'(BD, 0) &= \Lambda(C) \cup \Lambda(D) = CD, \\
 \delta'(BD, 1) &= \emptyset, \\
 \delta'(CD, 0) &= \Lambda(D) = D, \\
 \delta'(CD, 1) &= \Lambda(B) = BD, \\
 \delta'(D, 0) &= \Lambda(D) = D,
 \end{aligned}$$

$$\delta'(D, 1) = \emptyset.$$

Por tanto, los estados de  $M'$  son:  $ABD, ABCD, BD, CD, D$  y  $\emptyset$ . Como  $D$  es el único estado aceptador de  $M$ , los estados aceptadores de  $M'$  son  $ABD, ABCD, BD, CD$  y  $D$  (es decir, todos los estados salvo  $\emptyset$ ).

(3) Representamos al estado  $ABD$  por 0, al estado  $ABCD$  por 1, al estado  $BD$  por 2, al estado  $CD$  por 3 y al estado  $D$  por 4. Como  $\emptyset$  es un estado de error, no hace falta representarlo. 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 return false;
      break;
      case 1:
      if (c == '1') q = 2;
      break;
      case 2:
      if (c == '0') q = 3; else return false;
      break;
      case 3:
      if (c == '0') q = 4; else q = 2;
      break;
      case 4:
      if (c == '1') return false;
      break;}
    c = entrada.charAt(++i); }
  return true; }
```

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; }

```

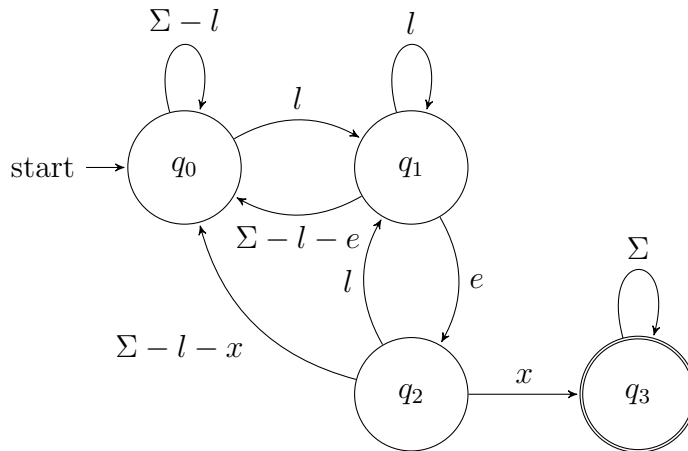
Problema 8. (a) Definir un autómata determinista que determine si un texto de caracteres contiene el patrón "lex".

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

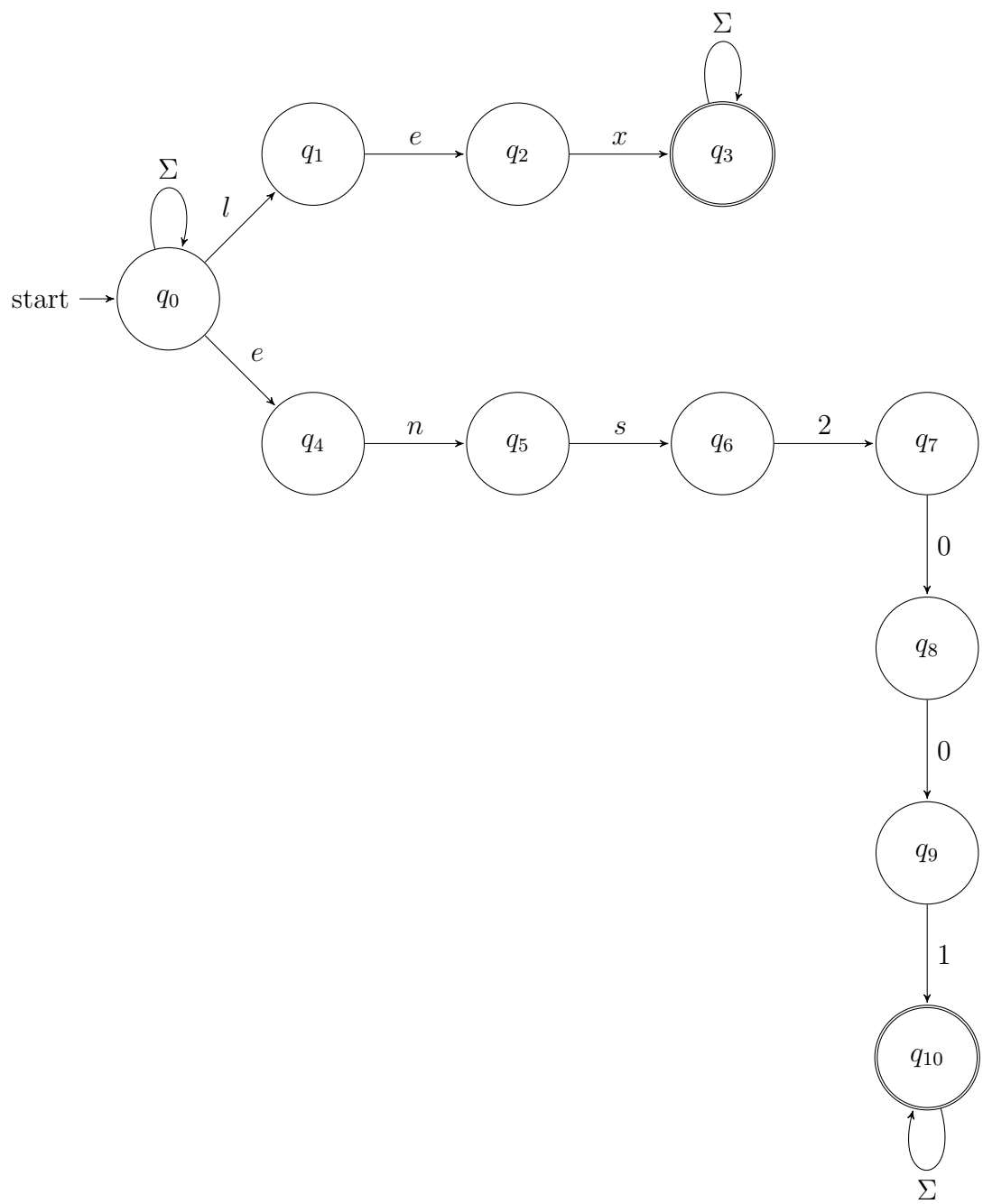
(c) Programar en Java el autómata determinista definido en (a).

**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).

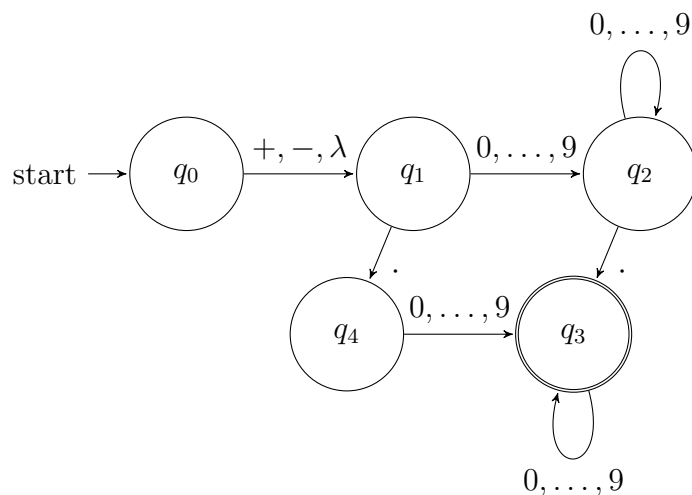
```
public boolean simular (String entrada)
{ int q = 0, i = 0;
  char c = entrada.charAt(0);
  while (c != '$')
  { switch(q)
    { case 0:
      if (c == 'l') q = 1;
      break;
      case 1:
      if (c == 'e') q = 2; else if ((c != 'e') && (c != 'l')) q = 0;
      break;
      case 2:
      if (c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

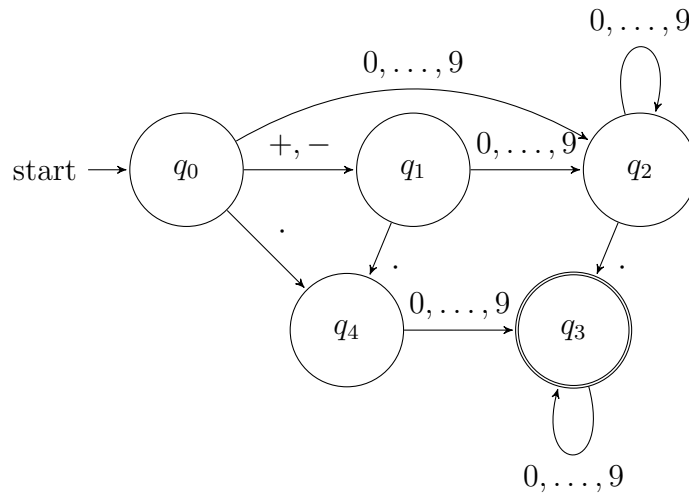
**Solución:**

(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  $\lambda$ .



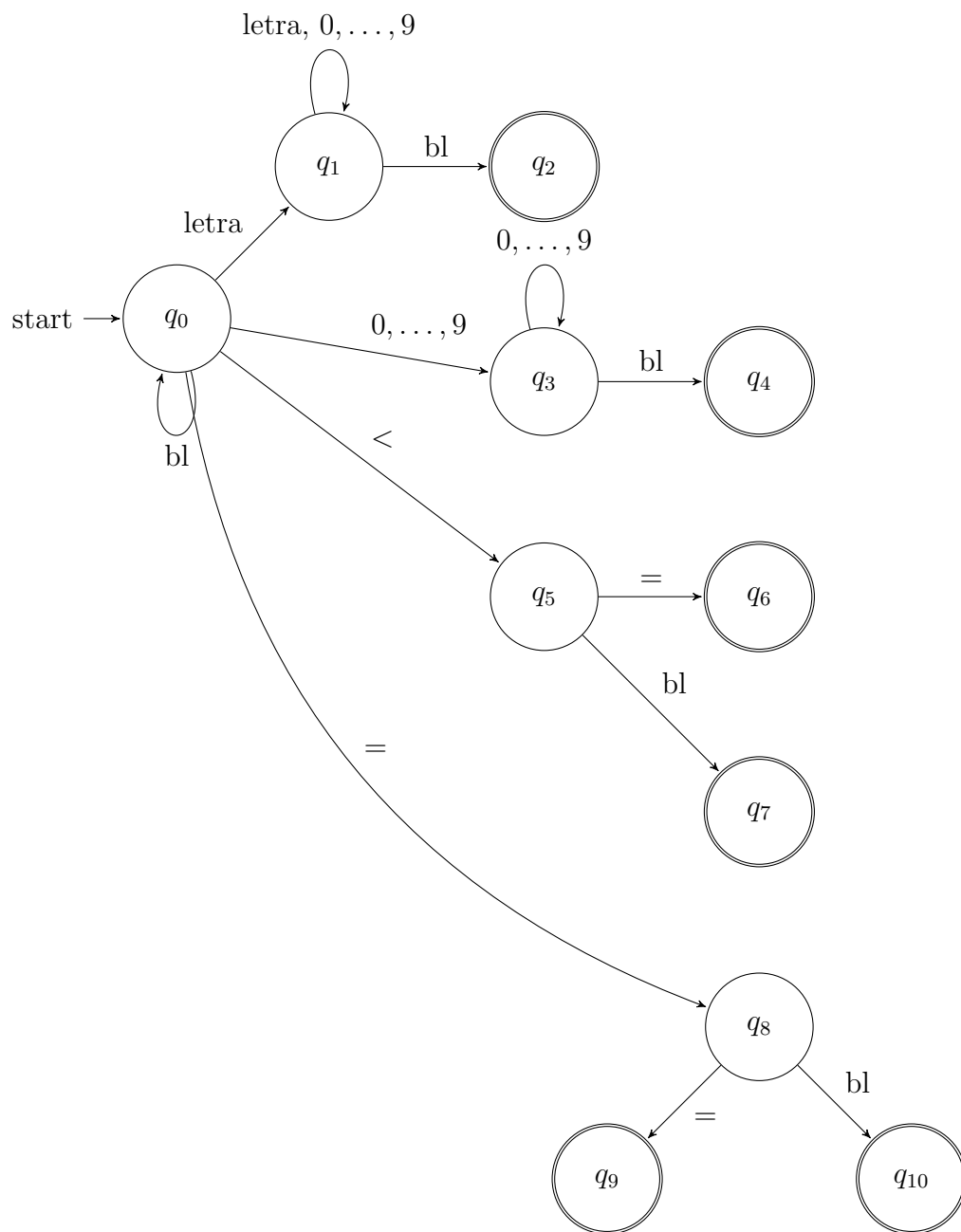
Ahora, podemos definir el autómata determinista equivalente, simplemente añadiendo a este último autómata  $M_2$  un estado de error, al cual llegaremos desde cualquier otro estado cuando entre un carácter distinto a los indicados en las transiciones del gráfico del autómata  $M_2$ . El programa asociado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