Capítulo 1: Gramáticas y Lenguajes Formales.

01.- Dado el Lenguaje L = { λ, cb, abbc, babac } determinar:

a) Los posibles prefijos, infijos y posfijos de cada cadena.
b) El Alfabeto Σ.
c) La longitud de cada cadena de L.
d) El contenido del conjunto L^2.
e) ¿L es un Lenguaje finito o infinito?

a.

λ	Prefijos:	λ
	Posfijos:	λ
	Infijos:	λ

cb	Prefijos:	λ, c, cb
	Posfijos:	λ, b, cb
	Infijos:	λ, c, b, cb

abbc	Prefijos:	λ, a, ab, abb, abbc
	Posfijos:	λ, c, bc, bbc, abbc
	Infijos:	λ, a, b, c, ab, bc, abb, bbc, abbc

babac	Prefijos:	λ, b, ba, bab, baba, babac
	Posfijos:	λ, c, ac, bac, abac, babac
	Infijos:	λ, a, b, c, ba, ab, ac, bab, aba, bac, baba, abac, babac

b. $\sum = \{a, b, c\}$

c. $|\lambda| = 0$

$|cb| = 2$

$|ab^2c| = 4$

$|(ba)^2c| = 5$

d. $L^2 = \{\lambda, cb, ab^2c, (ba)^2c, (cb)^2, cbab^2c, cb(ba)^2c, ab^2c^2b, (ab^2c)^2, ab^2c(ba)^2c, (ba)^2c^2b, (ba)^2cab^2c, [(ba)^2c]^2\}$

e. Finito.

02.- Dados los Lenguajes $L_1 = \{ a, aa, ab, aba \}$ y $L_2 = \{ \lambda, b, ab, bab, bbb \}$ evaluar:

a) $L_1 \cup L_2$	d) $L_2 - L_1$
b) $L_1 \cap L_2$	e) $L_1 - L_2$
c) $L_1 \bullet L_2$	f) $L_2 \cup L_1 \bullet L_2$

a. $L_1 \cup L_2 = \{\lambda, a, b, aa, ab, aba, bab, bbb\}$

b. $L_1 \cap L_2 = \{ab\}$

c. $L_1 \cdot L_2 = \{a, ab, a^2b, (ab)^2, ab^3, a^2, a^3b, a(ab)^2, a^2b^3, ab^2, ab^2ab, ab^4, aba, aba^2b, (ab)^3, (ab)^2b^2\}$

d. $L_2 - L_1 = \{\lambda, b, bab, bbb\}$

e. $L_1 - L_2 = \{a, aa, aba\}$

f. $L_2 \cup L_1 \cdot L_2 = \{\lambda, a, b, ab, bab, b^3, a^2b, (ab)^2, ab^3, a^2, a^3b, a(ab)^2, a^2b^3, ab^2, ab^2ab, ab^4, aba, aba^2b, (ab)^3, (ab)^2b^2\}$

03.- Dado el Lenguaje $L = \{ \lambda, a, ab, c \}$, determinar el contenido de los conjuntos Σ, L^0, L^1, L^2, L^+ y L^*.

$\sum = \{a, b, c\}$

$L^0 = \{\lambda\}$

$L^1 = \{\lambda, a, b, c\}$

$L^2 = \{\lambda, a, ab, c, a^2, a^2b, ac, aba, (ab)^2, abc, ca, cab, c^2\}$

$L^+ = \{L^1 \cup L^2 \cup \ldots \cup L^\infty\}$

$L^* = \{L^0 \cup L^1 \cup \ldots \cup L^\infty\}$

04.- Considérese el Lenguaje L^*, surgido a partir de $L = \{a, b\}$ con un Alfabeto $\Sigma = \{ a, b \}$.

a) ¿Cuántas palabras de ese L^* tienen longitud de 2, de 3, de 4 y de n, respectivamente?

b) Evaluar lo mismo que en el inciso anterior, pero ahora con $L = \{ a, ab, bc \}$ si $\Sigma = \{ a, b, c \}$.

a. $|s| = 2$; Palabras = 4 ; $\{a^2, ab, ba, b^2\}$

$|s| = 3$; Palabras = 8 ; $\{a^3, a^2b, aba, \ldots, b^3\}$

$|s| = 4$; Palabras = 16 ; $\{a^4, a^3b, a^2ba, \ldots, b^4\}$

$|s| = n$; Palabras = 2^n ; $\{a^n, \ldots, b^n\}$

b. $|s| = 2$; Palabras = 3 ; $\{a^2, ab, bc\}$

$|s| = 3$; Palabras = 5 ; $\{a^3, a^2b, abc, aba, bca\}$

$|s| = 4$; Palabras = 11 ; $\{a^4, a^3b, a^2bc, aba^2, bca^2, a^2ba, abca, ab^2c, bcab, (ab)^2, (bc)^2\}$

$|s| = n$; Palabras = ¿?

05.- Considérese al Lenguaje $L = \{ a, ab, bb \}$. Las cadenas aabab, babbbab, abbabab, abbaababbb, ¿son palabras de un L^n?

$a\text{-}ab\text{-}ab \in L^3$

$babbbab \notin L^n$

$a\text{-}bb\text{-}ab\text{-}ab \in L^4$

$a\text{-}bb\text{-}a\text{-}ab\text{-}ab\text{-}bb \in L^6$

06.- Dado el Lenguaje $L = \{ aa, aba, baa \}$, responder:

a) Las cadenas a^2ba^2, $(ba^3)^2$, $(ba^2)^3b^2a$, $ba^5(ba)^2a^3$ ¿son palabras del Lenguaje L^+?

b) ¿Qué características tendrían las cadenas de este L^+, expresadas en un Lenguaje natural?

a. $a^2ba^2 = aa\text{-}baa \in L^+$
$(ba^3)^2 = baa\text{-}aba\text{-}aa \in L^+$
$(ba^2)^3b^2a = baa\text{-}baa\text{-}baa\text{-}\cancel{bba} \notin L^+$
$ba^5(ba)^2a^3 = baa\text{-}aa\text{-}aba\text{-}baa\text{-}aa \in L^+$

b. Las cadenas de este L^+ :
- Terminan en "a".
- No existen "b" consecutivas.
- Contienen una cantidad para de "a" a partir de 2.
- Contienen el doble o más de "a" que de "b".

07.- Describir en un Lenguaje natural las cadenas que contiene cada uno de los siguientes Lenguajes Formales:

a) $L = \{ 1^*0 \}$
b) $L = \{ 1^*00^* \}$
c) $L = \{ 1^20^* \} \cup \{ 0^21^* \}$
d) $L = \{ (1, 00)^* \}$
e) $L = \{ (0^+1)^* \}$
f) $L = \{ (0,1)^1 \bullet (0,1)^* 00 \}$

a. Cadenas de bits que tienen una cantidad de 1's indeterminada y siempre contienen un cero al final.

b. Cadenas de bits que contienen al menos un cero y todos estos van al final de la cadena.

c. Cadenas de bits que inician con dos 1's seguidos de cualquier cantidad de 0's o bien comienzan con dos 0's seguidos de cualquier cantidad de 1's.

d. Cadenas de bits que cuando aparecen 0's estos vienen consecutivos y en cantidades pares.

e. Cadenas de bits que no contienen 1's consecutivos, además de siempre comenzar con 0 y terminar con 1, excepto λ.

f. Cadenas de bits cuya longitud es mayor que dos y siempre termina en 0.

08.- Determinar en cuáles de los siguientes Lenguajes está contenida la cadena 1011.

a) $L = \{ 10^*1^* \}$
b) $L = \{ 0^* (11,10)^* \}$
c) $L = \{ (10)^* 101 \}$
d) $L = \{ 1^* 01 (0,1)^1 \}$
e) $L = \{ (11)^* (10)^* \}$
f) $L = \{ (11)^*, (10)^* \}$
g) $L = \{ 1^+ (01)^* 1^* \}$
h) $L= \{(1,00) \bullet (01,0) \bullet 1^*\}$
i) $L = \{ 10, 11 \}^3$
j) $L = \{ 0, 1 ,10 \}^n$

a. $1011 \in L$
b. $1011 \in L$
c. $1011 \notin L$
d. $1011 \in L$
e. $1011 \notin L$
f. $1011 \notin L$
g. $1011 \in L$
h. $1011 \in L$
i. $1011 \notin L$
j. $1011 \in L$

09.- Considerando a $\Sigma = \{ a, b, c \}$, ¿son equivalentes las parejas de expresiones formales de cada inciso?

a) $((a \cup b) c)^*$ y $(ac \cup bc)^*$.
b) $b (ab \cup ac)$ y $(ba \cup ba) (b \cup c)$.
c) $(a, b)^*$ y (a^*, b^*).
d) $((a \cup b) a)^*$ y $(a \cup b)^* a^*$.

a. Equivalente.
b. Equivalente.
c. No equivalente.
d. No equivalente.

10.- Especificar la expresión formal que corresponda a cada uno de los siguientes Lenguajes sobre $\Sigma = \{ a, b \}$:

a) El Lenguaje de las cadenas que inician con a^2 **o** que finalizan con b^2.
b) El Lenguaje de las cadenas que inician con a^2 **y** que finalizan con b^2.

a. $L = \{a^2(a,b)^*\} \cup \{(a,b)^*b^2\}$

b. $L = \{a^2(a,b)^*b^2\}$

11.- Sea G = {(A, B, C), (a, b, c), (A → aaA, A → bB, B → cCb, B → cb, C → bbC, C → bb), (A)}.

Establecer cuáles de las siguientes cadenas pertenecen a L (G).

a) aabcb.
b) abbcb.
c) bcbbbbb.
d) aabccbbb.
e) aaaabcbb.
f) aaaabcbbb.

a. $\underline{A} \rightarrow aa\underline{A} \rightarrow aab\underline{B} \rightarrow aabcb \in L(G)$

b. $abbcb \notin L(G)$

c. $\underline{A} \rightarrow b\underline{B} \rightarrow bc\underline{C}b \rightarrow bcbb\underline{C}b \rightarrow bcbbbbb \in L(G)$

d. $aabccbbb \notin L(G)$

e. $aaaabcbb \notin L(G)$

f. $\underline{A} \rightarrow aa\underline{A} \rightarrow aaaa\underline{A} \rightarrow aaaab\underline{B} \rightarrow aaaabc\underline{C}b \rightarrow aaaabcbbb \in L(G)$

12.- Dada G =({ a, b, c }, { +, - }, { a → a +, a → - c -, b - → +, b c → +, b → a + c, c - → λ, c → c c, a - c → + + }, { a }).

a) ¿Qué observaciones se le pueden hacer a G?
b) Caracterizar la Gramática anterior.
c) Encontrar 6 cadenas del Lenguaje L(G).

a. Observaciones:

- En la cadena no se producen "b".
- Jamás se aplica la regla $a-c \rightarrow ++$
- La regla $c \rightarrow cc$ haría imposible seguir derivando la cadena (esta regla se puede usar, pero no se debe).
- La gramática se puede simplificar como:
  $G = \{(a, c), (+, -), (a \rightarrow a+,\ a \rightarrow -c-,\ c- \rightarrow \lambda), (a)\}$

b. $L(G) = \{-+^*\}$

$\underline{a} \rightarrow \underline{a}+ \rightarrow \ldots \rightarrow \underline{a}+^* \rightarrow -\underline{c-}+^* \rightarrow -+^*$

c. $L(G) = \{-,\ -+,\ -+^2,\ -+^3,\ -+^4,\ -+^5,\ \ldots,\ -+^n\}$

13.- Caracterizar la Gramática con composiciones P = {S → AB, A → aA, A → a, B → Bb, B → b} y σ_0 = { S }.

$L(G) = \{a^+b^+\}$

$\underline{S} \rightarrow \underline{A}\ \underline{B}$

$\underline{A} \rightarrow a\underline{A} \rightarrow aa\underline{A} \rightarrow \ldots \rightarrow a^*\underline{A} \rightarrow a^*a = a^+$

$\underline{B} \rightarrow \underline{B}b \rightarrow \underline{B}bb \rightarrow \ldots \rightarrow \underline{B}b^* \rightarrow bb^* = b^+$

14.- Sean N = { S, A, B }, T = { a, b }, σ_0 = { S }. Para cada inciso caracterizar el Lenguaje generado por la Gramática señalada y mencionar si existe algún error en las especificaciones y cómo se puede corregir.

a) P = { $S \to A \mid B,\ A \to abA \mid c,\ B \to ccB \mid ab$ }.
b) P= { $S \to AA \mid B,\ A \to aaA \mid aa,\ B \to bB \mid b$ }.
c) P = { $S \to AB \mid AA,\ A \to aB \mid ab,\ B \to b$ }.
d) P = { $S \to AS \mid aA,\ A \to a,\ B \to ba$ }.

a. La gramática no es válida, por lo tanto no se podría caracterizar. Sin embargo, si T = {a, b, c}, entonces la gramática es válida y se podría caracterizar.

$L(G) = \{(ab)^*c\} \cup \{(c^2)^*ab\}$

$\underline{S} \to \underline{A} \mid \underline{B}$
$\underline{S} \to \underline{A} \to ab\underline{A} \to abab\underline{A} \to \ldots \to (ab)^*\underline{A} \to (ab)^*c$
$\underline{S} \to \underline{B} \to (c^2)\underline{B} \to (c^2)(c^2)\underline{B} \to \ldots \to (c^2)^*\underline{B} \to (c^2)^*ab$

b.
$$L(G) = \left.\begin{array}{c} \{(a^2)^*a^4\} \\ \{(a^2)^+a^2\} \\ \{(a^2)^+(a^2)^+\} \\ \{a^{2m+4} \mid m \in \mathbb{N}_0\} \end{array}\right\} \cup \left\{\begin{array}{c} \{b^+\} \\ \{b^n \mid n \in \mathbb{N}\} \end{array}\right.$$

c. $L(G) = \{ab^2, (ab)^2\} = \{ab^2\} \cup \{(ab)^2\}$

$\underline{S} \to \underline{A}\,\underline{B} \mid \underline{A}\,\underline{A}$
$\underline{S} \to \underline{A}\,B \to a\underline{BB} \to abb \to ab^2$
$\underline{S} \to \underline{A}\,\underline{A} \to abab \to (ab)^2$

d. $L(G) = \{a^*a^2\} = \{a^+a\}$

$\underline{S} \to A\underline{S} \to AA\underline{S} \to \ldots \to A^*\underline{S} \to A^*a\underline{A} \to \underline{A}^*aa \to a^*a^2$

15.- Caracterizar la Gramática con P = { $A \to xAz,\ A \to yAz,\ A \to \lambda$ } y σ_0 = { A }. ¿Es la anterior una Gramática decidible?

$L(G) = \{(x, y)^n z^n \mid n \in \mathbb{N}_0\}$
Si es decidible, porque $\lambda \in L(G)$

16.- Diseñar la Gramática que produce cada uno de los L(G) enunciados.

a) $L(G) = \{ 0^n1^n \mid n \in \mathbb{N} \}$.
b) $L(G) = \{ 0^n1^n \mid n \in \mathbb{N}_o \}$.
c) $L(G) = \{ 0^m1^n \mid m, n \in \mathbb{N} \}$.

a. $G = \{(A), (0, 1), (A\to 0A1, A\to 01), (A)\}$
b. $G = \{(A), (0, 1), (A\to 0A1, A\to\lambda), (A)\}$
c. $G = \{(A), (0, 1), (A\to 0A, A\to A1, A\to 01), (A)\}$

17.- Diseñar la Gramática Formal que produce el Lenguaje $L = \{(ab)^*c^2\}$.

$G = \{(A), (a, b, c), (A\to abA, A\to cc), (A)\}$

18.- Diseñar para cada inciso una Gramática que genere todas las cadenas de bits con las siguientes características:

a) Con igual cantidad de 0 que de 1.
b) Con más 0 que 1.
c) Con desigual cantidad de 0 que de 1.

a. $G = \{(A, I, O), (0, 1), (A\to OIA \mid \lambda, OI\to IO, IO\to OI, I\to 1, O\to 0), (A)\}$
b. $G = \{(A, I, O), (0, 1), (A\to OIA \mid OA \mid O, OI\to IO, IO\to OI, I\to 1, O\to 0), (A)\}$
c. $G = \{(A, B, I, O), (0, 1), (A\to OIA \mid OA \mid O \mid OIB \mid IB \mid I, OI\to IO, IO\to OI, I\to 1, O\to 0), (A)\}$

19.- Diseñar una Gramática Regular que produzca el Lenguaje $L = \{ 01, 10 \}^+$ con $T = \{ 0, 1 \}$.

$G = \{(A, B, C), (0, 1), (A\to 0B \mid 1C, B\to 1, C\to 0, B\to 1A, C\to 0A), (A)\}$

20.- Diseñar la Gramática Regular que produce el Lenguaje $L = \{ a^*b \} \cup \{ a \}$ si el Alfabeto es $\Sigma = \{ a, b \}$.

$G = \{(A, B), (a, b), (A\to aB, B\to aB, B\to b, A\to a, A\to b), (A)\}$

21.- Demostrar que $L(G) = \{ 0^n1^n \mid n \varepsilon \mathbb{N} \}$ no es Lenguaje Regular.

No es un lenguaje regular, porque no tiene una gramática regular que lo genere. Es un lenguaje libre de contexto, porque la gramática que lo genera es libre de contexto: $G = \{(A), (0, 1), (A\to 0A1, A\to 01), (A)\}$

22.- Expresar al menos 5 de las reglas de producción más comunes de Lenguaje C en BNF y otras 4 reglas de Java o Pascal en Diagramas de Sintaxis.

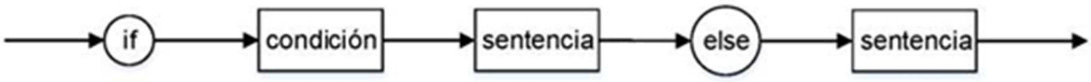
5 reglas de producción en C en BNF:

<instrucción si>::= if(<condición>) <sentencia>
<instrucción si_no>::= if(<condición>) <sentencia> else <sentencia>
<para>::= for(<asignación>; <condición>; <incremento>) <sentencia>
<mientras>::= while(<condición>) <sentencia>
<hacer>::= do <sentencia> while(<condición>);

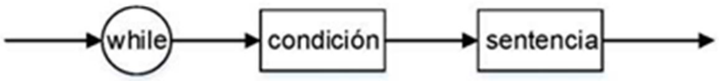
4 reglas de Java en diagrama de sintaxis:

Instrucción si:

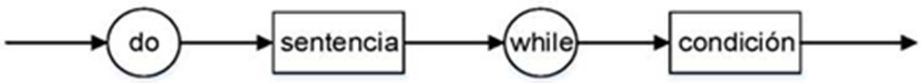
Instrucción si_no:



Mientras:



Hacer:



23.- Diseñar las composiciones en BNF para una Gramática que genere todos los números reales o flotantes válidos, en este caso sin tomar en cuenta el tipo de Gramática a emplear.

P = {<real>::= <real c/signo> | <real s/signo>, <real c/signo>::= <signo> <real s/signo>, <signo>::= + | –, <real s/signo>::= <secuencia de dígitos> . <secuencia de dígitos> | .<secuencia de dígitos>, <secuencia de dígitos>::= <dígito> <secuencia de dígitos> | <dígito>, <dígito>::=0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9}

24.- Diseñar una Gramática Regular en BNF para cada inciso, que produzca los siguientes tokens:

a) Identificadores válidos en Lenguaje C.
b) Números enteros decimales sin signo.
c) Números enteros decimales (con o sin signo).
d) Números reales decimales (con o sin signo).
e) Números enteros hexadecimales sin signo en ensamblador para microprocesadores de Intel.

a. G = {(<id>, <resto id>), (letra, _, dígito), (<id>::= letra <resto id> | letra | _<resto id> | _, <resto id>::= letra <resto id> | dígito <resto id> | _<resto id> | letra | dígito | _), (<id>)}

b. G = {(<entero s/signo>), (dígito), (<entero s/signo>::= dígito <entero s/signo> | dígito), (<entero s/signo>)}

c. G = {(<entero>, <entero s/signo>), (signo, dígito), (<entero>::= signo <entero s/signo> | digito <entero s/signo> | dígito, <entero s/signo>::= dígito <entero s/signo> | dígito), (<entero>)}

d. G = {(<real>, <real s/signo>, <entero s/signo>), (signo, . , dígito), (<real>::= signo <real s/signo> | dígito <real s/signo> | .<entero s/signo>, <real s/signo>::= dígito <real s/signo> | .<entero s/signo>, <entero s/signo>::= dígito <entero s/signo> | dígito), (<real>)}

e. G = {(<hexadecimal>, <resto hex>), (dígito, letra, hache), (<hexadecimal>::= dígito <resto hex>, <resto hex>::= dígito <resto hex> | letra <resto hex> | hache), (<hexadecimal>)}

25.- Dada la Gramática con las composiciones del conjunto

P = { <número> ::= <entero> | <real>,
<entero> ::= <dígito> | <entero> <dígito>,
<real>::= <entero>.<entero> |
<entero>.<entero>E<exp> | <entero> E <exp>,
<exp> ::= <entero> | <signo> <entero>,
<signo> ::= + | -,
<dígito> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 }

en donde <número> es el símbolo inicial; especificar qué inconvenientes tiene la Gramática, respecto a la producción de números válidos según los criterios empleados en el desarrollo de compiladores actuales.

Inconvenientes de la gramática:

- No es regular.
- No genera números con signo al inicio.
- No se genera una cadena que empiece con punto.
- No se aceptan las cadenas con "E" minúscula.

26.- Empleando la misma Gramática del inciso anterior evaluar tanto la derivación por la izquierda como por la derecha, para las cadenas 0.27 y 6E-3.

0.27 por la izquierda:

<número>::= <real>::= <entero>.<entero>::= <dígito>.<entero>::= 0.<entero>::= 0.<entero> <digito>::= 0.<dígito> <dígito>::= 0.2<dígito>::= 0.27

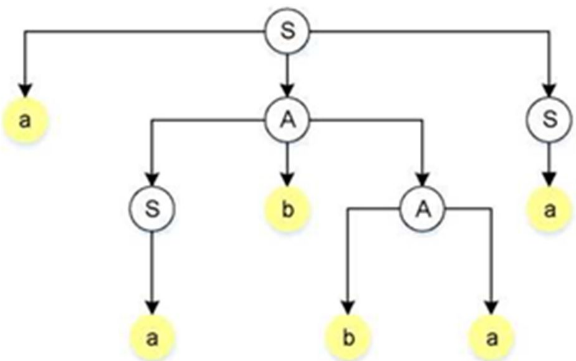
0.27 por la derecha:
<número>::= <real>::= <entero>.<entero>::= <entero>.<entero> <dígito>::= <entero>.<entero> 7::= <entero>.<dígito> 7::= <entero>.27::= <dígito>.27::= 0.27

6E−3 por la izquierda:
<número>::= <real>::= <entero> E <exp>::= <dígito> E <exp>::= 6E <exp>::= 6E <signo> <entero>::= 6E−<entero>::= 6E−<dígito>::= 6E−3

6E−3 por la derecha:
<número>::= <real>::= <entero> E <exp>::= <entero> E <signo> <entero>::= <entero> E <signo> <dígito>::= <entero> E <signo> 3::= <entero> E−3::= <dígito> E−3::= 6E−3

27.- Demostrar con un árbol de derivación, si esto fuera posible, que la cadena $a^2b^2a^2$ es una cadena del Lenguaje producido por la Gramática con P = { S → aAS | a, A → SS | SbA | ba }, con símbolo inicial S.

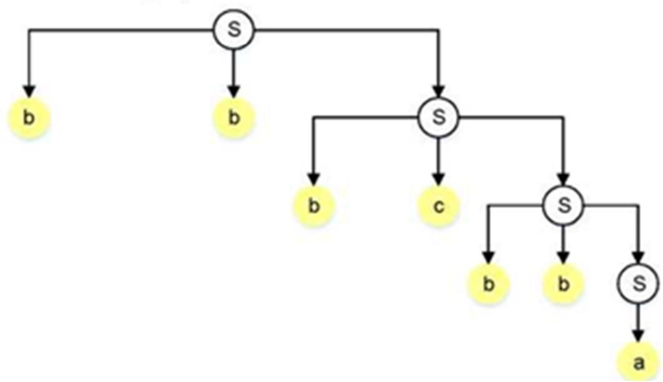
$a^2b^2a^2 \in L(G)$



28.- Dada la Gramática con G = ({ S }, { a, b, c }, { S → abS, S → bcS, S → bbS, S → a, S → cb }, { S }), construir los árboles de derivación para cada una de las cadenas:

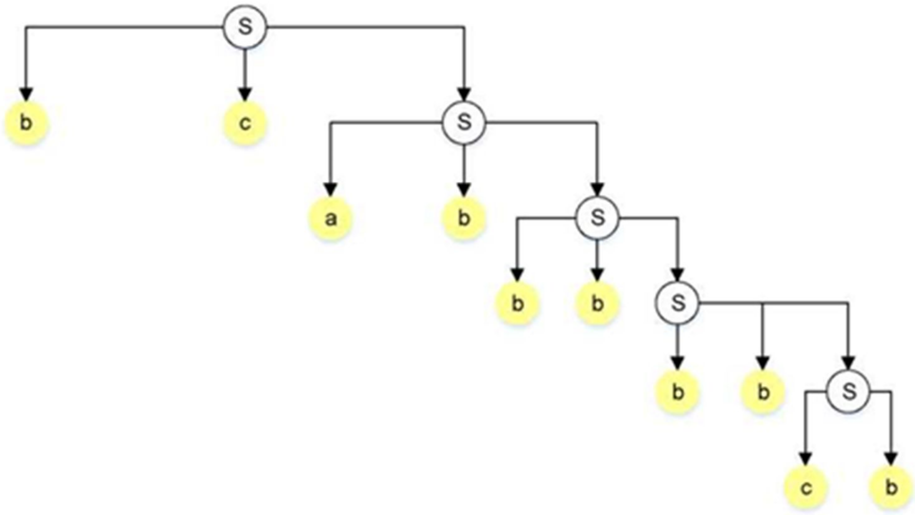
a) b^3cb^2a
b) ab^2ac^2ba
c) $bcab^5cb$
d) ab^2c^2ab
e) bcb^5ca
f) $(ab)^3b^4cb$

a. $b^3cb^2a \in L(G)$



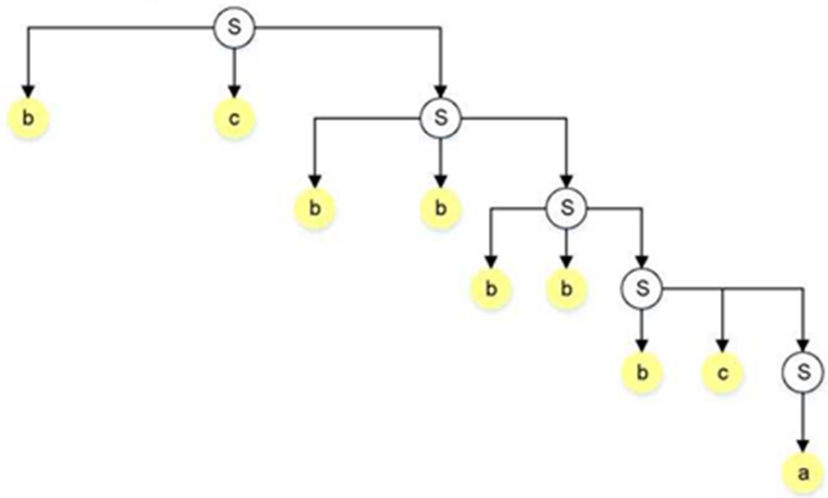
b. $ab^2ac^2ba \notin L(G)$

c. $bcab^5cb \in L(G)$

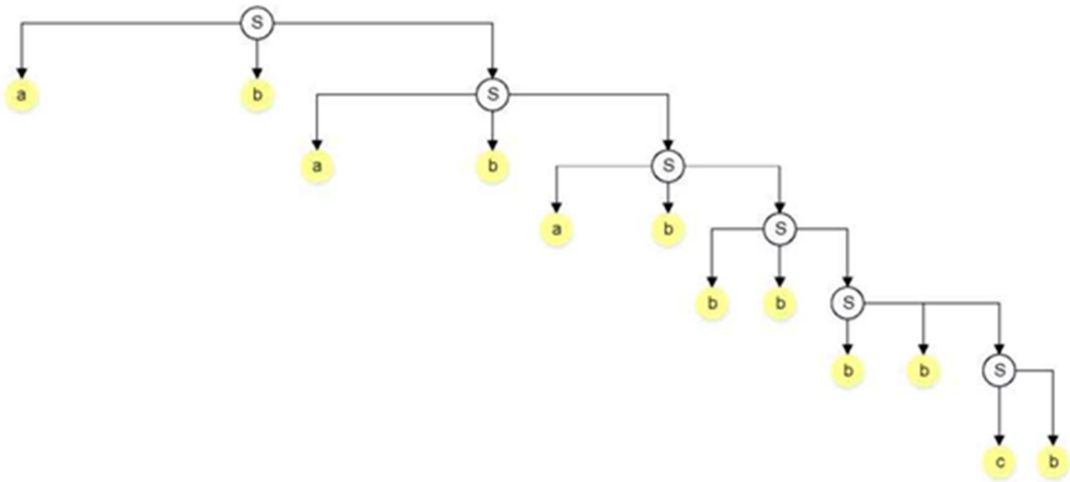


d. $ab^2c^2ab \notin L(G)$

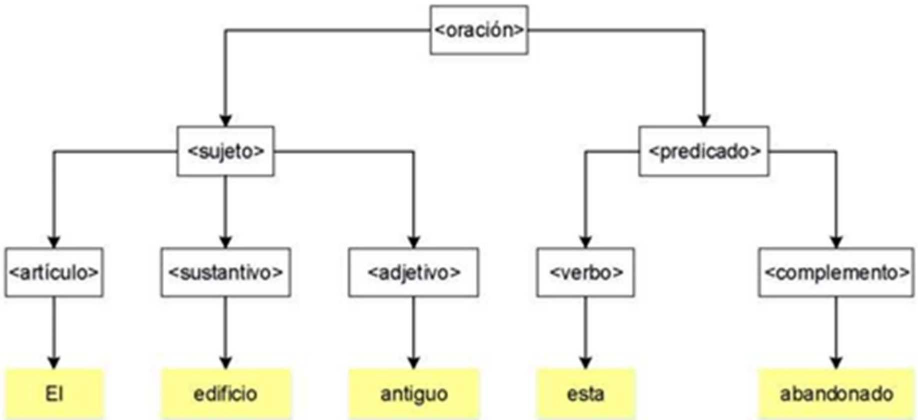
e. $bcb^5ca \in L(G)$



f. $(ab)^3b^4cb \in L(G)$



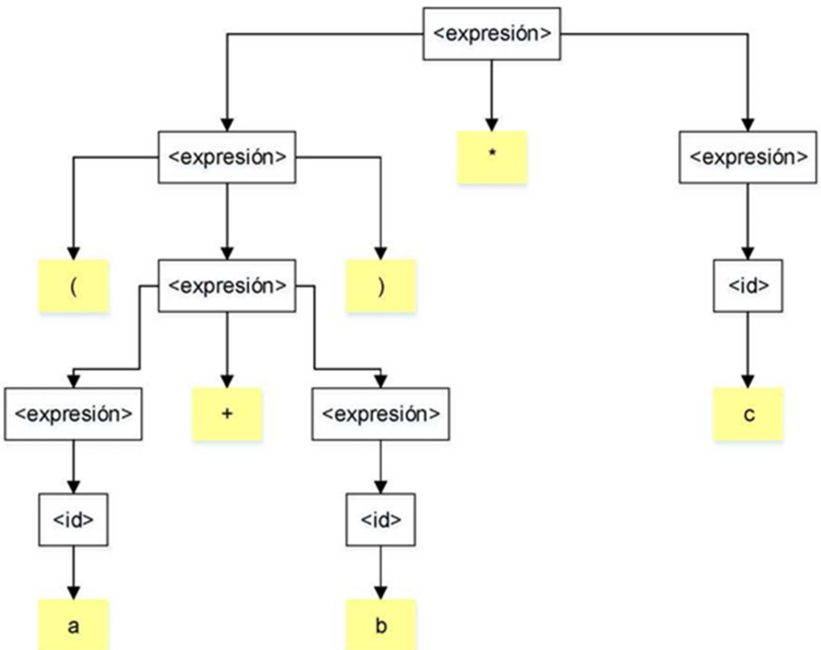
29.- Desarrollar una oración o sentencia válida de un Lenguaje natural, como el castellano, por medio de un árbol de derivación y utilizando las reglas de producción adecuadas, aprendidas durante la primaria.



NOTA: El verbo debe corresponder a la naturaleza del sustantivo y ser conjugado.

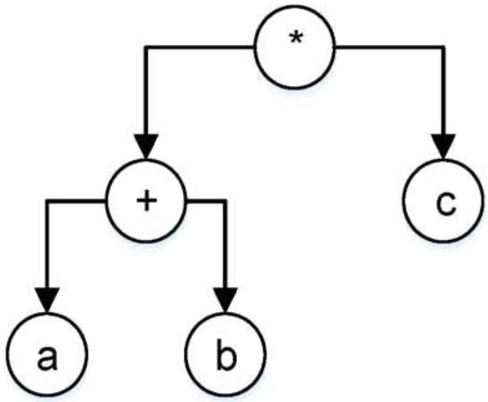
30.- Diseñar el árbol de derivación para la expresión tomada del código de un programa ***(a + b) * c.*** ¿Cuáles serían su árbol de análisis sintáctico y su árbol sintáctico?
Las composiciones a emplear deben ser deducidas por el alumno en base a sus conocimientos sobre programación.

Árbol de análisis sintáctico:



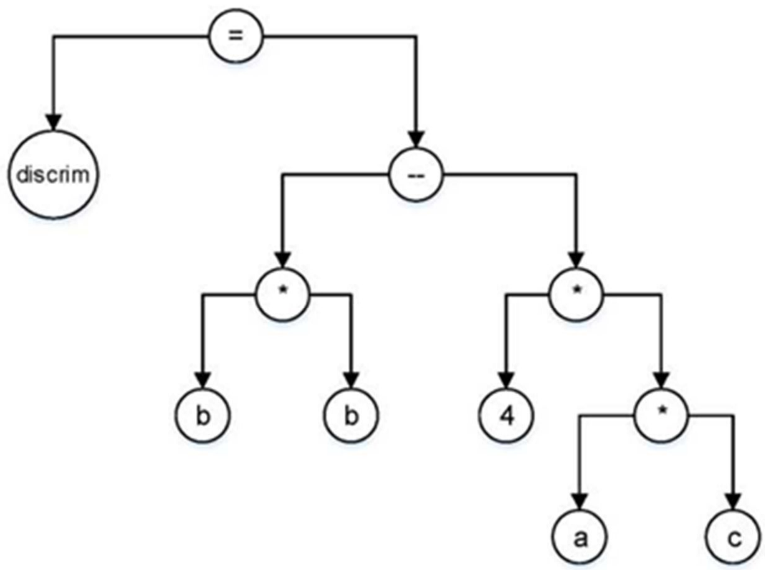
Donde:
<id> = <identificador>

Árbol sintáctico:

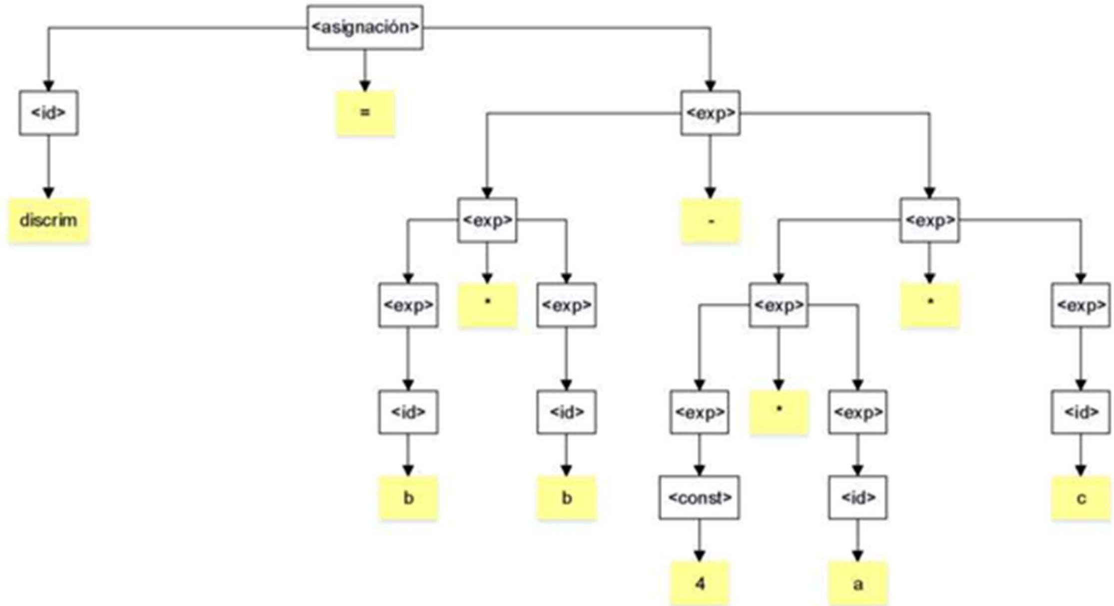


31.- Diseñar un árbol de sintaxis y otro de análisis sintáctico que correspondan a la siguiente cadena tomada de un programa en Lenguaje C: ***discrim* = *b* * *b* - 4 * *a* * *c*.** Definir las reglas usadas.

Árbol de sintaxis:



Árbol de análisis sintáctico:



Donde:

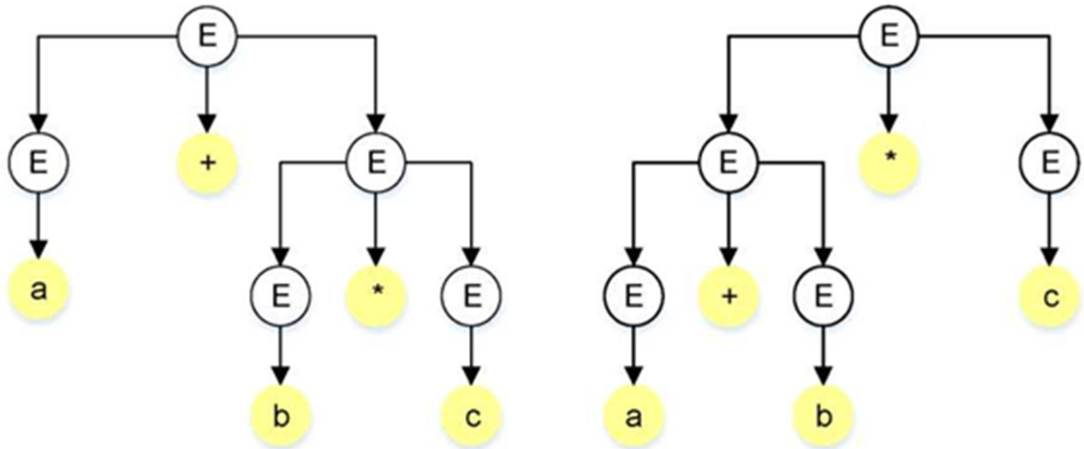
<exp> = <expresión>

<const> = <constante>

<id> = <identificador>

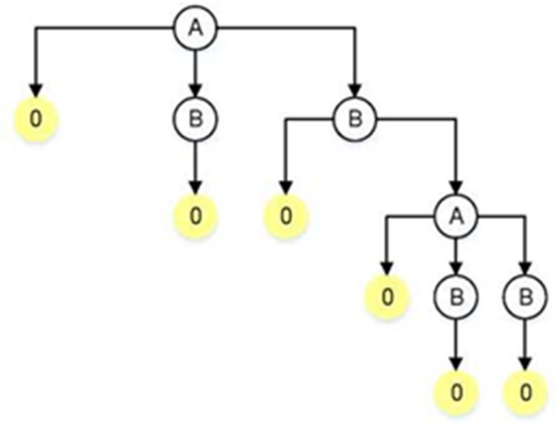
32.- Determinar si G es ambigua o unívoca: G = ({ E }, { (,), +, *, id }, { E → E + E | E * E | (E) | id }, { E }). En este caso, como la variedad de identificadores válidos es muy grande, se considera en la teoría de compiladores, para generalizarlos, como si *id* fuera un terminal definido.

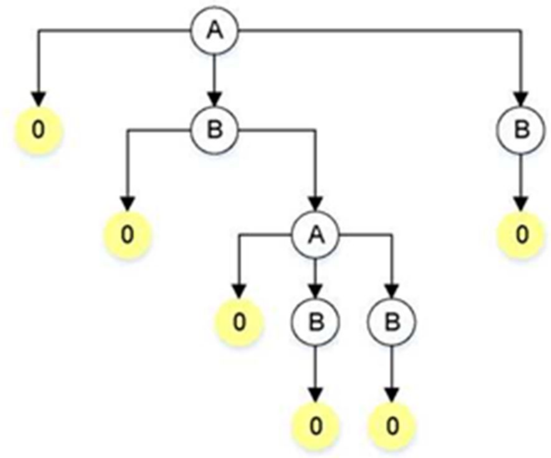
La gramática (G) es ambigua.



33.- Dada la Gramática con P = { A → 0 B B, B → 1 A | 0 A | 0 } con A como símbolo inicial, determinar si es ambigua o unívoca y justifíquese la respuesta.

La gramática (G) es ambigua porque, al igual que la gramática anterior, puede generar cadenas idénticas con diferentes arboles:

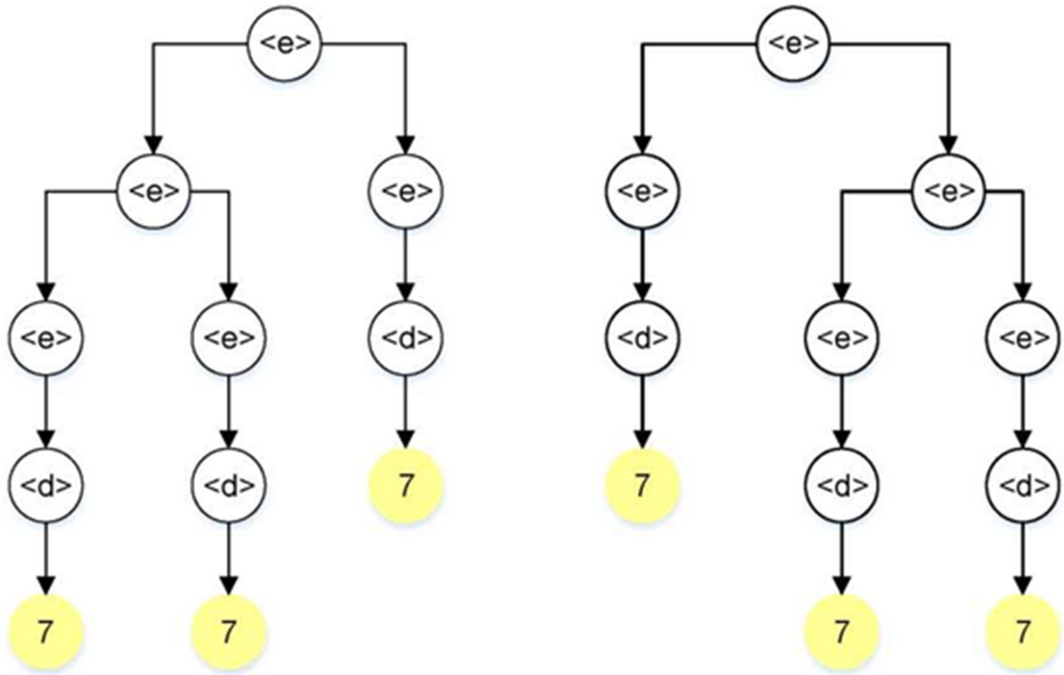




34.- Demostrar que la Gramática con P = { <entero s/signo> ::= <dígito> | <entero s/signo> <entero s/signo>, <dígito> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 } es ambigua y encontrar otra equivalente y no ambigua.

Se ha demostrado que la gramática (G) es ambigua porque puede generar cadenas iguales con árboles diferentes. Sin embargo, se puede generar una gramática equivalente y no ambigua, sus composiciones serían:

P = {<entero s/signo>::= <dígito> | <dígito> <entero s/signo>, <dígito>::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 }



Donde:

<e> = <entero s/signo>

<d> = <dígito>

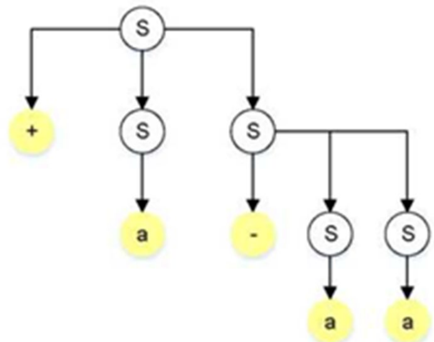
35.- Determinar si son ambiguas las siguientes Gramáticas:

a) P = { S → + S S | - S S | a }.

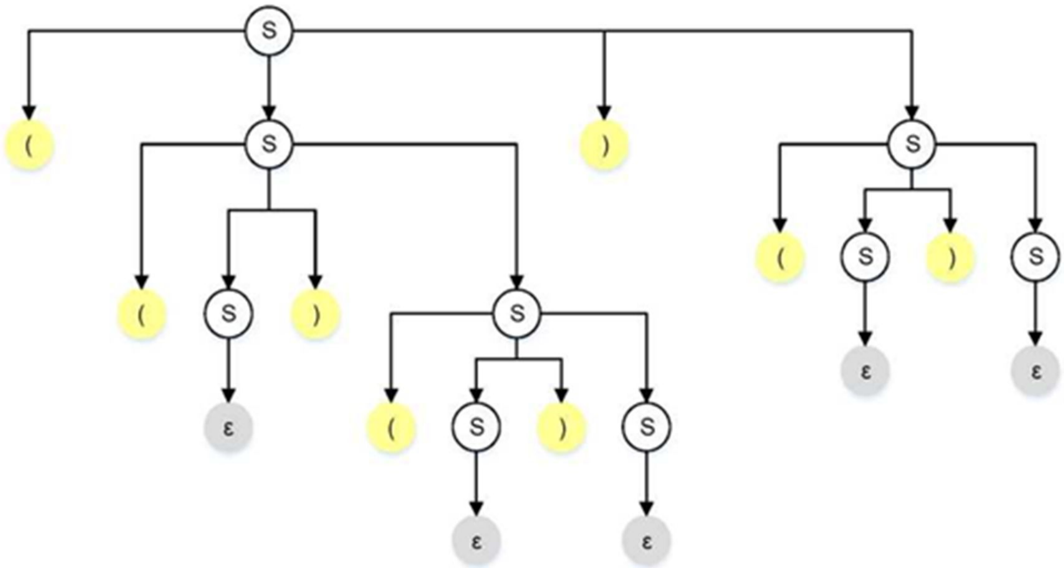
b) P = { S → (S) S | ε }. //ε es cadena vacía

c) P = { S → S ^ S | a }.

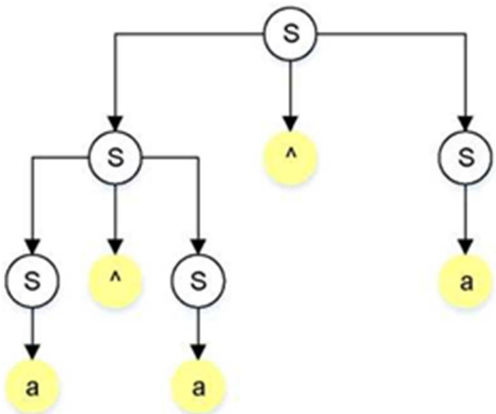
a. En la notación prefija y posfija las expresiones no son ambiguas y como la gramática genera expresiones en notación prefija podemos decir que es univoca

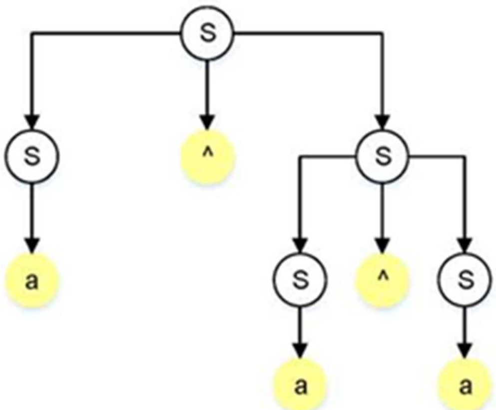


b. Esta gramática (G) nos ayuda a crear parejas de paréntesis bien anidados. Es univoca, porque no existen dos cadenas iguales que se puedan formar con diferentes árboles.



c. La gramática (G) es ambigua.





36.- Diseñar las reglas de producción para un compilador que evalúe sin que exista ambigüedad la exponenciación múltiple (a^b^c^d ...) y considerando la asociatividad.

En la exponenciación la asociatividad es por la derecha:
P = { <exponenciación>::= <id> ^ <id> | <id> ^ <exponenciación>, <id>::= letra <resto id> | letra | _<resto id> | _, <resto id>::= letra <resto id> | dígito <resto id> | _<resto id> | letra | dígito | _ }

37.- Diseñar una Gramática Libre de Contexto en la cual se jerarquicen las cuatro operaciones aritméticas básicas, además de incluir los paréntesis y considerando también su asociatividad. Llámese <expresión> al símbolo inicial.

G = ({<expresión>, <término>, <factor>}, {+, -, *, /, (,), id, constante}, {<expresión>::= <expresión> + <término> | <expresión> - <término> | <término>, <término>::= <término> * <factor> | <término> / <factor> | <factor>, <factor>::= (<expresión>) | id | constante}, {<expresión>})

38.- Convertir la Gramática Libre de Contexto a su Forma Normal de Chomsky (CNF):

P = { A → xy | Bx | xEx, D → xx | yB | xE,
B → zC | DE | E, E → EAx | yCy,
C → DC | xyF | FA | y, F → AC | B }.

P = {A→XY | BX | XG, B→ZC | DE | EI | YJ, C→DC | XH | FA | y, D→XX | YB | XE, E→EI | YJ, F→AC | ZC | DE | EI | YJ, G→EX, H→YF, I→AX, J→CY, X→x, Y→y}

39.- Convertir a la Forma Normal de Chomsky (CNF) equivalente la siguiente Gramática Libre de Contexto:

N = { A, B, C, D, E, F } T = { x, y } σ_0 = { A }

P = {	A→xy,	B→DEF,	C→y,	E→FA,
	A→Bx,	C→DC,	D→xx,	E→yCy,
	A→xEx,	C→xyF,	D→yB,	F→AC,
	B→zC,	C→FA ,	D→xE,	F→B }.

G = ({A, B, C, D, E, F, G, H, I, J, X, Y}, {x, y}, {A→XY | BX | XG, B→ZC | DH, C→DC | XI | FA | y, D→XX | YB | XE, E→FA | YJ, F→AC | ZC | DH, G→EX, H→EF, I→YF, J→CY, X→x, Y→y}, {A})

CAPITULO 3: Máquinas de Estado Finito.

41.- Dibujar el diagrama de transición de la Máquina de Estado Finito que corresponde a cada inciso, además de expresar los seis conjuntos que forman la representación de la tupla para M.

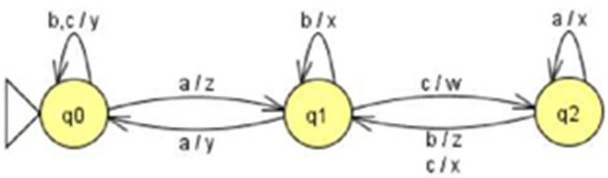
a)

S \ I	f: a	f: b	f: c	g: a	g: b	g: c
▶▶ q_0	q_1	q_0	q_0	z	y	y
q_1	q_0	q_1	q_2	y	x	w
q_2	q_2	q_1	q_1	x	z	x

b)

S \ I	f: x	f: y	f: z	g: x	g: y	g: z
▶▶ q_0	q_1	q_3	q_2	a	b	a
q_1	q_0	q_2	q_1	d	a	c
q_2	q_2	q_0	q_3	c	b	b
q_3	q_2	q_3	q_1	d	c	b

a.



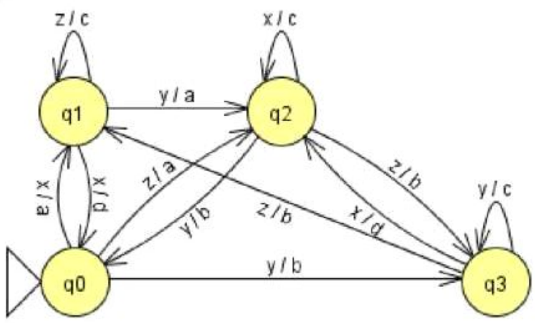
I = {a, b, c}
O = {w, x, y, z}
S = {q_0, q_1, q_2}

f = {f(q_0, a) = q_1, f(q_0, b) = q_0, f(q_0, c) = q_0, f(q_1, a) = q_0, f(q_1, b) = q_1, f(q_1, c) = q_2, f(q_2, a) = q_2, f(q_2, b) = q_1, f(q_2, c) = q_1}

g = {g(q_0, a) = z, g(q_0, b) = y, g(q_0, c) = y, g(q_1, a) = y, g(q_1, b) = x, g(q_1, c) = w, g(q_2, a) = x, g(q_2, b) = z, g(q_2, c) = x}

σ_0 = {q_0}

b.



$I = \{x, y, z\}$
$O = \{a, b, c, d\}$
$S = \{q_0, q_1, q_2, q_3\}$

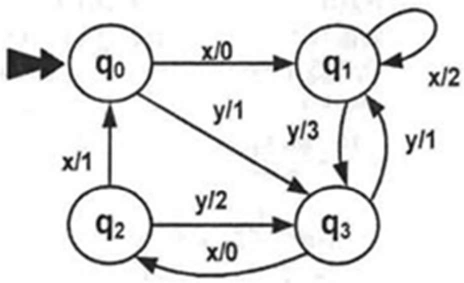
$f = \{f(q_0, x) = q_1, f(q_0, y) = q_3, f(q_0, z) = q_2, f(q_1, x) = q_0, f(q_1, y) = q_2, f(q_1, z) = q_1,$
$f(q_2, x) = q_2, f(q_2, y) = q_0, f(q_2, z) = q_3, f(q_3, x) = q_2, f(q_3, y) = q_3, f(q_3, z) = q_1\}$

$g = \{g(q_0, x) = a, g(q_0, y) = b, g(q_0, z) = a, g(q_1, x) = d, g(q_1, y) = a, g(q_1, z) = c,$
$g(q_2, x) = c, g(q_2, y) = b, g(q_2, z) = b, g(q_3, x) = d, g(q_3, y) = c, g(q_3, z) = b\}$

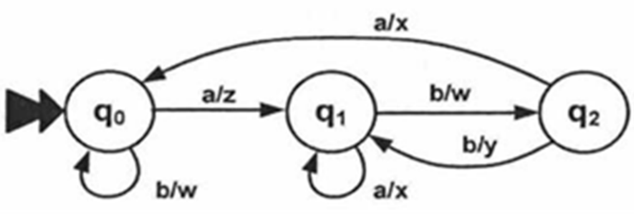
$\sigma_0 = \{q_0\}$

42.- Determinar cuáles son los conjuntos que determinan la descripción formal para las siguientes Máquinas de Estado Finito, además de las tablas de transición para cada inciso.

a)



b)



a.

I = {x, y}

O = {0, 1, 2, 3}

S = {q_0, q_1, q_2, q_3}

f = {f(q_0, x) = q_1, f(q_0,y) = q_3, f(q_1, x) = q_1, f(q_1, y) = q_3, f(q_2, x) = q_0, f(q_2, y) = q_3, f(q_3, x) = q_2, f(q_3, y) = q_1}

g = {g(q_0, x) = 0, g(q_0, y) = 1, g(q_1, x) = 2, g(q_1, y) = 3, g(q_2, x) = 1, g(q_2, y) = 2, g(q_3, x) = 0, g(q_3, y) = 1}

σ_0 = {q_0}

	f		g	
S \ I	x	y	x	y
q_0	q_1	q_3	0	1
q_1	q_1	q_3	2	3
q_2	q_0	q_3	1	2
q_3	q_2	q_1	0	1

b.

I = {a, b}

O = {w, y, z}

S = {q_0, q_1, q_2}

f = {f(q_0, a) = q_1, f(q_0, b) = q_0, f(q_1, a) = q_1, f(q_1, b) = q_2, f(q_2, a) = q_0, f(q_2, b) = q_1}

g = {g(q_0, a) = z, g(q_0, b) = w, g(q_1, a) = x, g(q_1, b) = w, g(q_2, a) = x, g(q_2, b) = y}

σ_0 = {q_0}

	f		g	
S \ I	a	b	a	b
q_0	q_1	q_0	z	w
q_1	q_1	q_2	x	w
q_2	q_0	q_1	x	y

43.- Establecer las transiciones (estados y cadenas de salida) que resultan al introducir las siguientes cadenas de entrada, referidas a las Máquinas de los problemas 41 y 42.

a) babccab en 41a)
b) zxyzxzyz en 41b)
c) xyxyyxyxxy en 42a)
d) bbabbbaaba en 42b).

a.

Cadena de entrada:		b	a	b	c	c	a	b
Cadena de salida:		y	z	x	w	x	y	y
Estado actual:	q_0	q_0	q_1	q_1	q_2	q_1	q_0	q_0

b.

Cadena de entrada:		z	x	y	z	x	z	y	z
Cadena de salida:		a	c	b	a	c	b	c	b
Estado actual:	q_0	q_2	q_2	q_0	q_2	q_2	q_3	q_3	q_1

c.

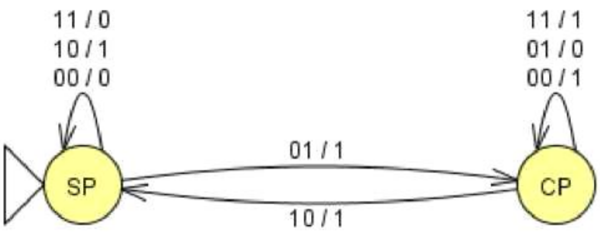
Cadena de entrada:		x	y	x	y	y	x	y	x	x	y
Cadena de salida:		0	3	0	2	1	2	3	0	1	1
Estado actual:	q_0	q_1	q_3	q_2	q_3	q_1	q_1	q_3	q_2	q_0	q_3

d.

Cadena de entrada:		b	b	a	b	b	b	a	a	b	a
Cadena de salida:		w	w	z	w	y	w	x	z	w	x
Estado actual:	q_0	q_0	q_0	q_1	q_2	q_1	q_2	q_0	q_1	q_2	q_0

44.- Diseñar una Máquina de Estado Finito que funcione como restador binario en serie y otra que se comporte como un multiplicador binario en serie, para operar con cadenas de bits de la misma longitud. Justificar las respuestas.

- Restador binario en serie:



Donde:

SP = Sin préstamo.
CP = Con préstamo.

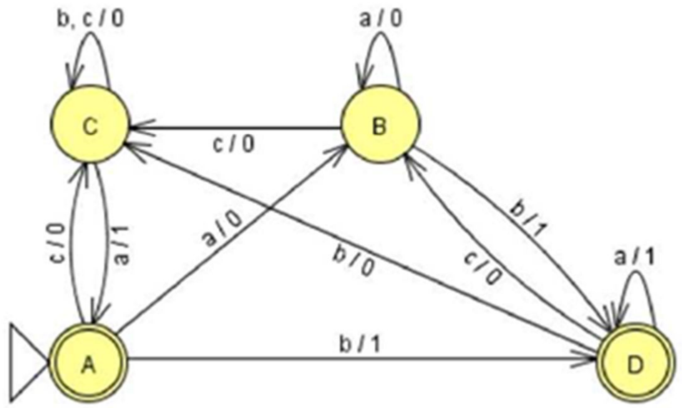
- Multiplicador binario en serie:

No es posible realizar la multiplicación binaria en serie porque:

- El resultado no es aritméticamente correcto.
- No hay estados en la multiplicación debido a que 0*0, 0*1, 1*0 siempre es 0, y 1*1 siempre es 1; no hay estados de sin préstamo o con préstamo, como en el restador binario en serie, ni de acarreo o sin acarreo, como en el sumador binario en serie.

45.- Diseñar una Máquina de Estado Finito, que también sea Autómata de Estado Finito, con las siguientes características:

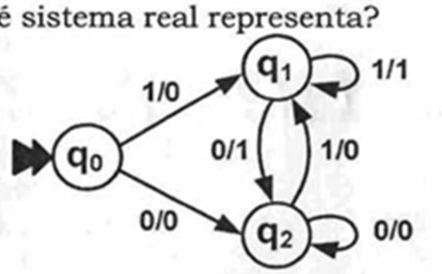
$I = \{a, b, c\}$, $S = \{A, B, C, D\}$, $\sigma_0 = \{A\}$.



46.- Dado el siguiente diagrama de transición determinar:

a) ¿Se puede transformar esta Máquina en un Autómata Finito?

b) ¿Qué sistema real representa?



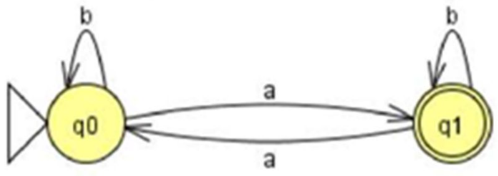
a. No, porque en los estados q_1 y q_2 dos de los tres arcos que inciden (entran) en ellos tienen salidas diferentes.

b. Representa un sistema SHR (SHift to Right) de lenguaje ensamblador, que hace un corrimiento de bits a la derecha.

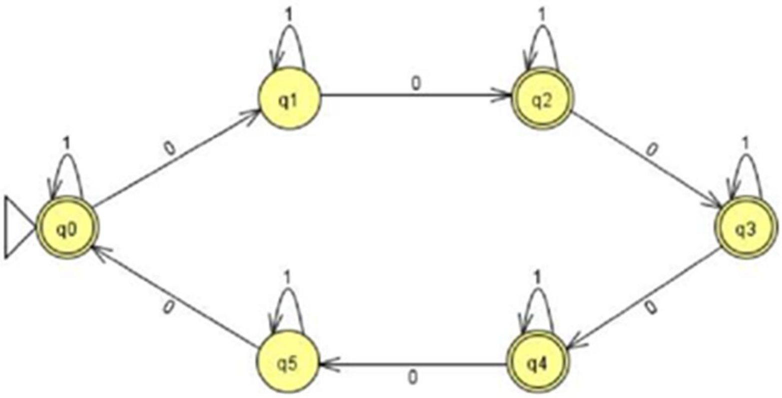
Cadena de entrada:		1	0	1	1	1	0	0	1	0	1
Cadena de salida:		0	1	0	1	1	1	0	0	1	0
Estado actual:	q_0	q_1	q_2	q_1	q_1	q_1	q_2	q_2	q_1	q_2	q_1

CAPITULO 4: Autómatas de Estado Finito.

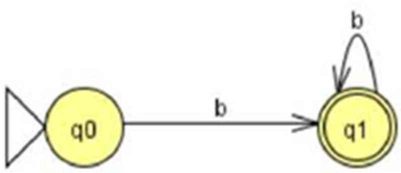
47.- Diseñar un Autómata Finito con I = { *a*, *b* } que acepte cadenas con una cantidad impar de letras *a*.



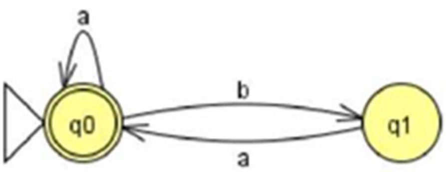
48.- Diseñar un Autómata de Estado Finito que acepte las cadenas de bits con una cantidad de 0 igual a un múltiplo de 2 ó de 3.



49.- Diseñar un Autómata Finito que acepte cadenas no nulas formadas con los símbolos del conjunto I = { *a*, *b*, *c* } y que no contengan ni *a* ni *c*.

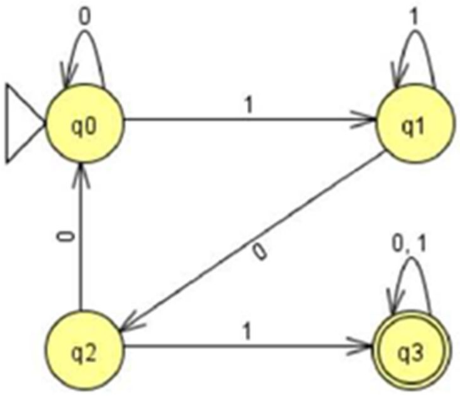


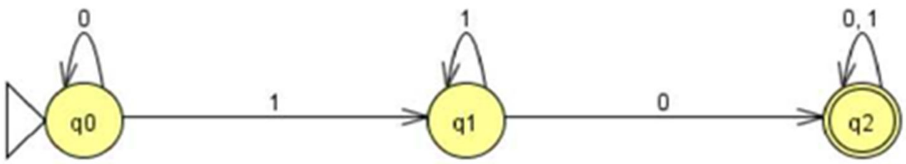
50.- Diseñar un Autómata Finito con I = { *a*, *b* } que acepte las palabras en las que siempre que aparezca una *b*, esté seguida necesariamente por una *a*.



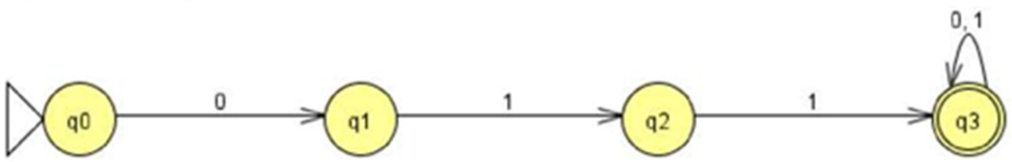
51.- Diseñar un Autómata de Estado Finito que acepte las cadenas de bits que contienen el infijo 10.

52.- Diseñar un Autómata de Estado Finito que acepte las cadenas de bits que contienen el infijo 101.





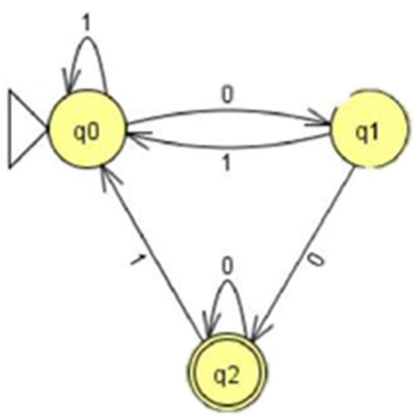
53.- Diseñar un Autómata Finito con I = { 0, *1* } que acepte los arreglos que inicien con 011.



54.- Diseñar un Autómata Finito que acepte todas las cadenas de bits que finalicen con 00 y mostrar el resultado por medio de las tres representaciones posibles para el modelo formal.

A = ({0, 1}, $\{q_0, q_1, q_2\}$, $\{q_2\}$, $\{f(q_0, 0) = q_1, f(q_0, 1) = q_0, f(q_1, 0) = q_2, f(q_1, 1) = q_0, f(q_2, 0) = q_2, f(q_2, 1) = q_0\}$, $\{q_0\}$)

	f	
S \ I	0	1
▷ q_0	q_1	q_0
q_1	q_2	q_0
q_2	q_2	q_0



55.- Diseñar el Autómata Finito que acepta cada unc de los siguientes Lenguajes, considerando I = { *a, b, c* }

a) $\varnothing$

b) $\{ \lambda \}$

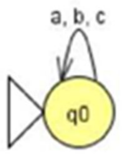
c) $\{ a, b, c \}$

d) $\{ (a, b, c)^* \}$

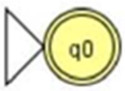
e) $\{ a^*b^+c^* \}$

f) $\{ bca^* \} \cup \{ ac^* \}$

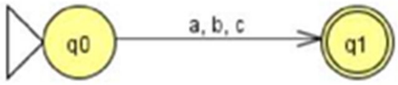
a.



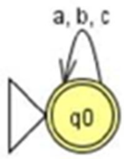
b.



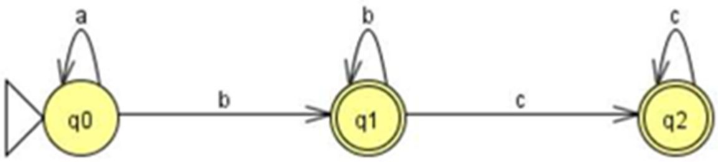
c.



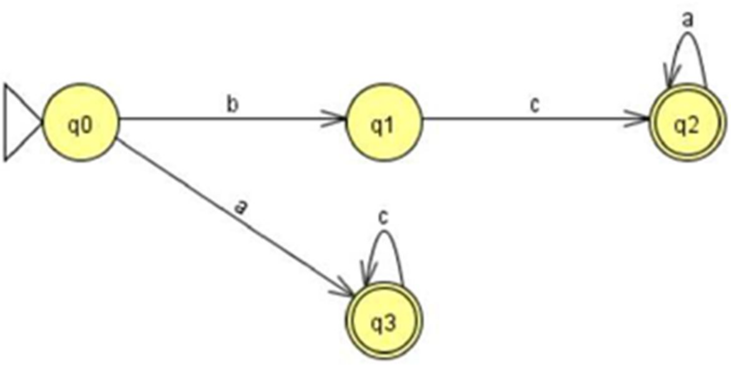
d.



e.



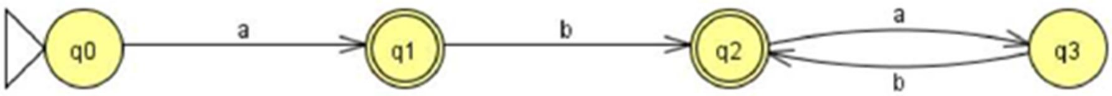
f.



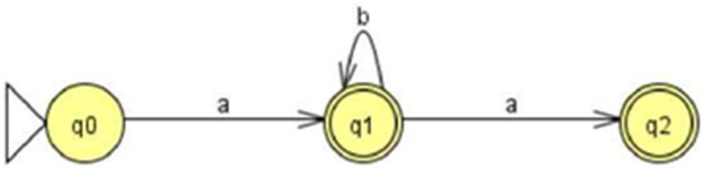
56.- Dado el conjunto I = { a, b } diseñar un diagrama de transición del Autómata Finito para cada uno de los Lenguajes expresados en los siguientes incisos:

a) $L = \{ (ab)^{+} \} \cup \{ a \}$.
b) $L = \{ ab^{*} \} \cup \{ ab^{*}a \}$.
c) $L = \{ a\,(a,b)^{*} \} \cup \{ b\,(a,b)^{*} \}$

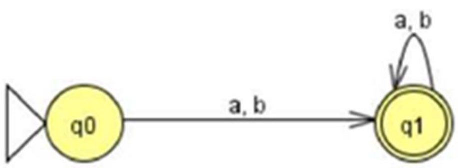
a.



b.



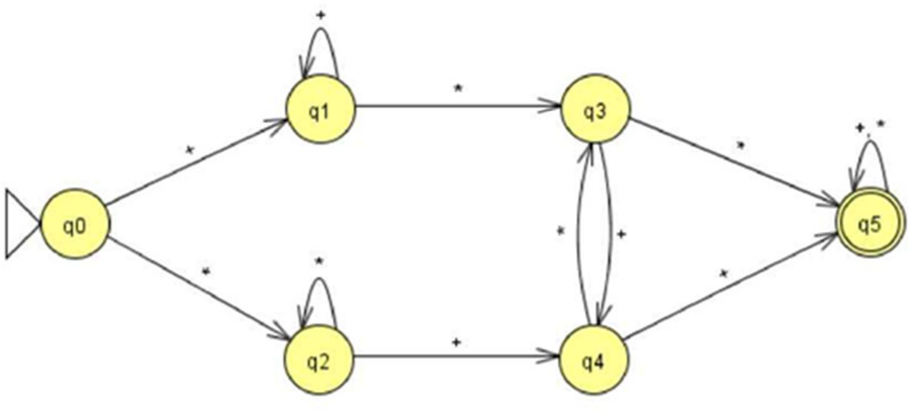
c.



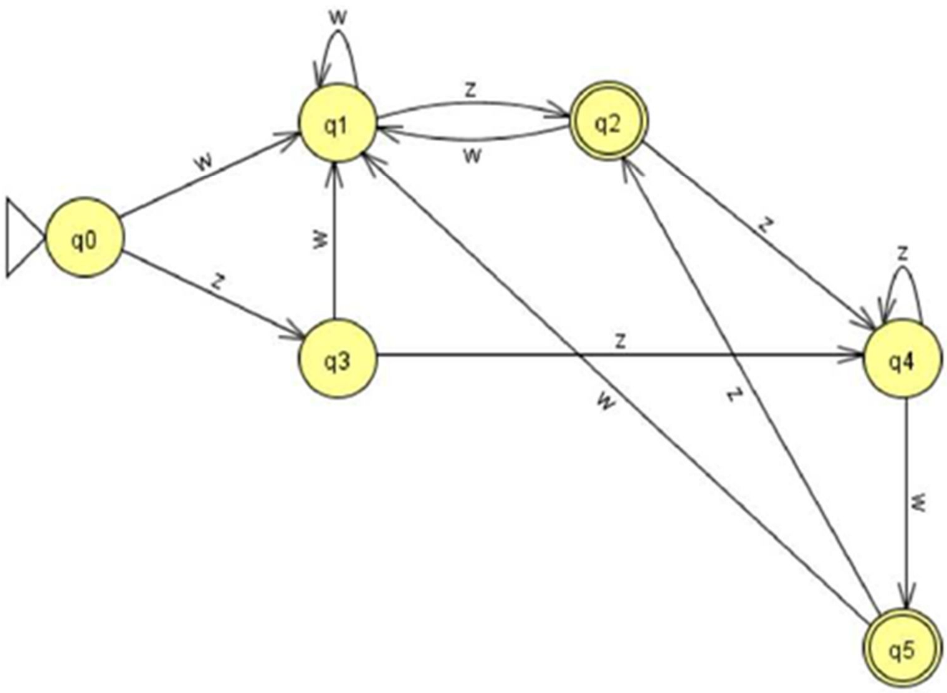
57.- Diseñar un Autómata Finito en cada inciso de acuerdo a las entradas y las cadenas aceptadas que se indican.

a) I = {+, *}. Cadenas que contienen +** ó *++.
b) I = {w, z}. Cadenas que terminen en wz o z^2w.

a.



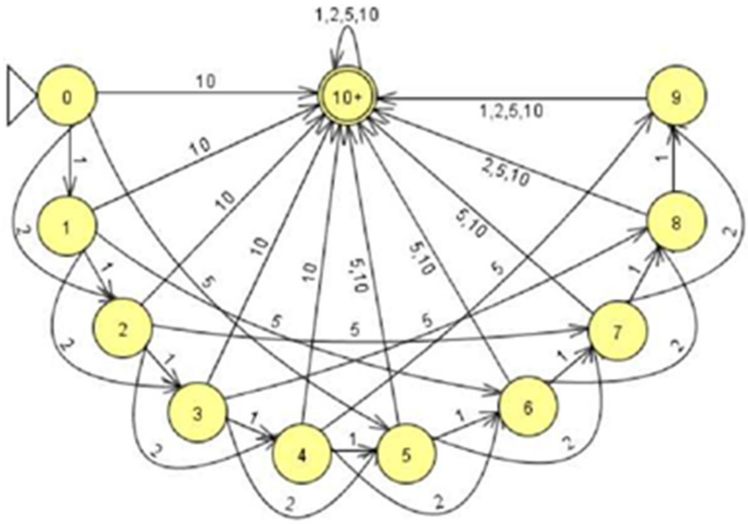
b.

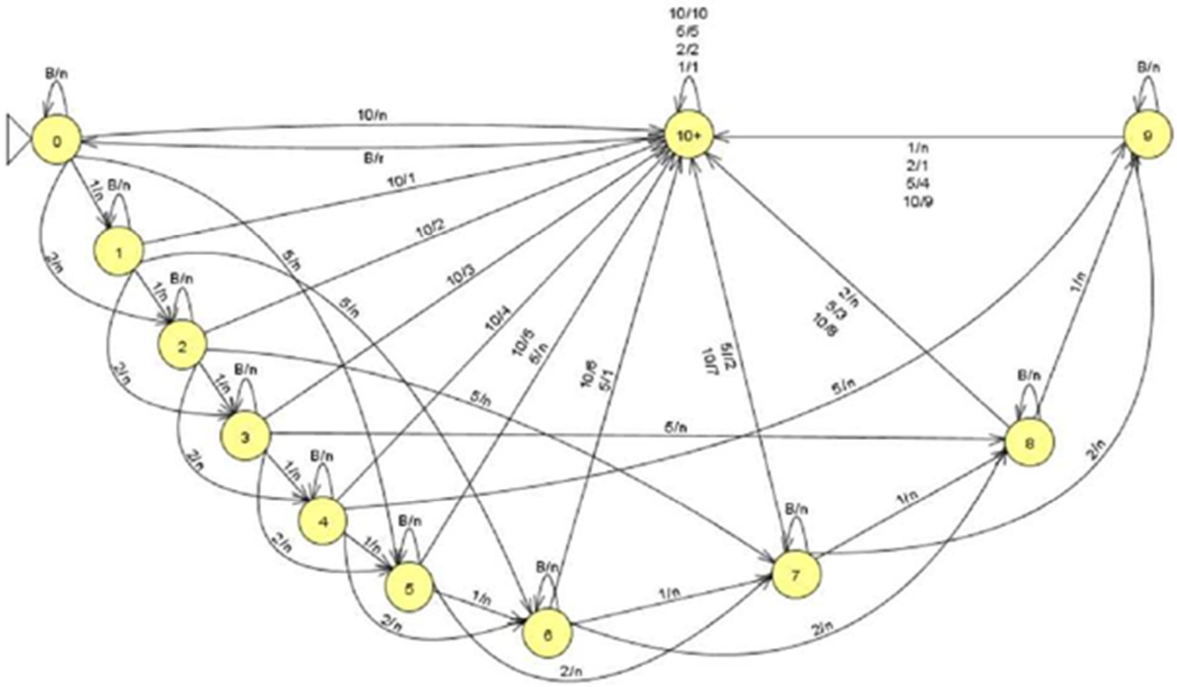


58.- Demostrar que no es posible construir un Autómata de Estado Finito que al recibir como entrada cualquier cadena de bits, acepte la cadena que contiene exactamente la misma cantidad de 0 que de 1.

Este lenguaje se define formalmente como $L = \{0^n1^n \mid n \in \mathbb{N}\}$; tal como se explica en el problema 21, es un lenguaje libre de contexto, por lo tanto, según la jerarquía de Chomsky un autómata de estado finito NO puede reconocer las cadenas de este lenguaje, sin embargo podrían ser reconocidas por un autómata de pila.

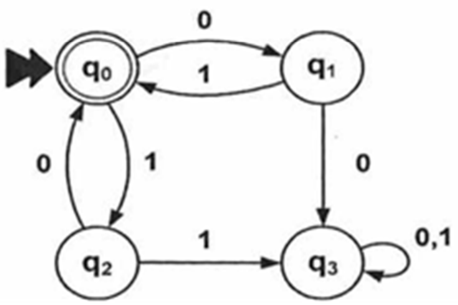
59.- Comparar entre el diseño de una máquina para venta de refrescos implementada con los modelos de Autómata y de Máquina de Estado Finito para descubrir la diferencia entre las dos herramientas matemáticas. Para fines didácticos considérese el caso de un refresco con un coste de \$10 y empleando monedas de 1\$, \$2, \$5 y \$10 como posibles entradas.





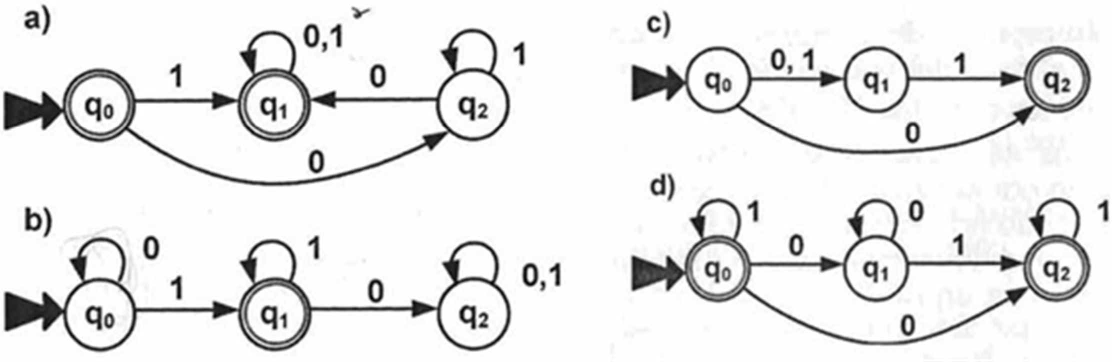
Nótese que en este caso, el modelo de la máquina de estados es más recomendable, debido a que se deben considerar varias salidas y no solamente la aceptación o rechazo de ciertas condiciones. Obsérvese que la máquina no tiene estados de aceptación o de no aceptación.

60.- ¿Qué cadenas acepta el siguiente Autómata Finito?



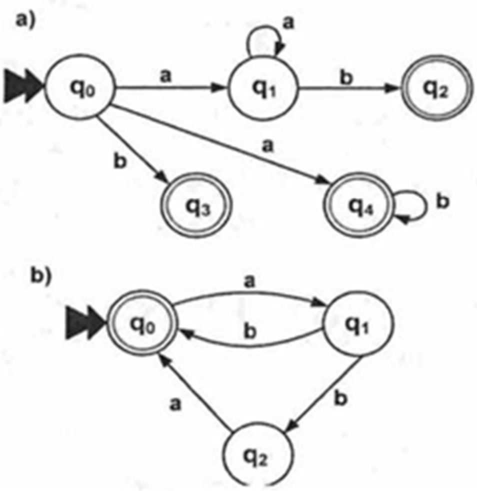
L = {01, 10}* = {(01, 10)*}

61.- Determinar el Lenguaje aceptado por cada uno de los siguientes Autómatas Finitos, observando que dos de ellos son No Determinísticos.



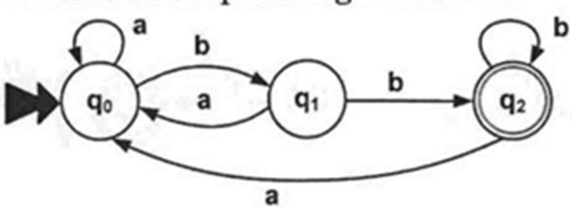
a. $L = \{\lambda\} \cup \{1(0, 1)^*\} \cup \{01^*0(0,1)^*\}$
b. $L = \{0^*1^+\}$
c. $L = \{0, 01, 1^2\}$
d. $L = \{1^*\} \cup \{1^*01^*\} \cup \{1^*0^+1^+\}$

62.- Caracterizar el Lenguaje aceptado por los siguientes Autómatas Finitos No Determinísticos:



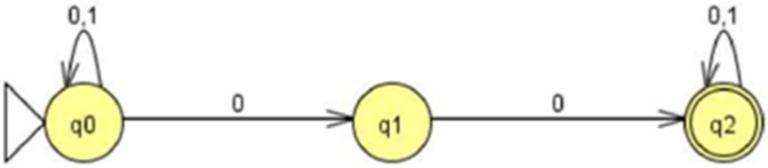
a. $L = \{a^+b, ab^*, b\} = \{a^*b, ab^*\}$
b. $L = \{ab, aba\}^*$

63.- ¿Qué cadenas acepta el siguiente AFD?

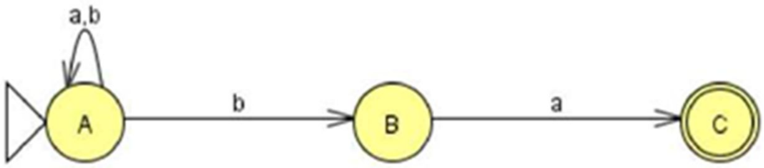


$L = \{(a, b)^*b^2\}$

64.- Demostrar que el Lenguaje de todas las cadenas de unos y ceros que tienen al menos dos ceros consecutivos es un Lenguaje Regular.

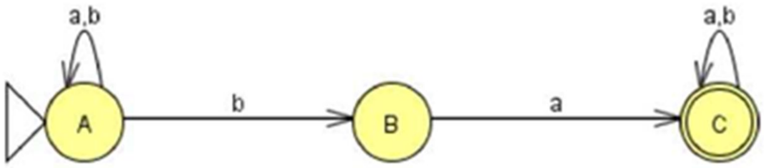


65.- Diseñar una Gramática Regular con T = { a, b } que genere todos los arreglos que finalicen con ba. El diseño se deberá hacer a partir de un Autómata Finito.



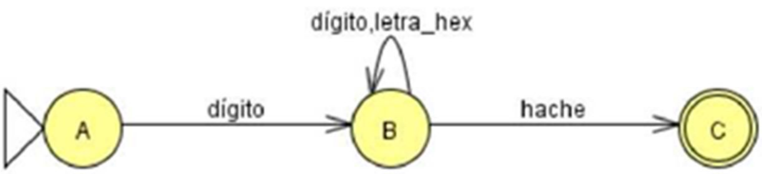
G = {(A, B, ~~C~~), (a, b), (A→Aa | bA | bB, ~~B→aC~~, B→a), (A)}

66.- Diseñar por medio de un Autómata Finito una Gramática Regular con T = { a, b } que genere todas las cadenas que contengan el infijo ba.



G = {(A, B, C), (a, b), (A→Aa | bA | bB, B→aC | a, C→aC | bC | b | a), (A)}

67.- Diseñar por medio de un Autómata Finito la Gramática Regular que produce todas las cadenas que representan los números enteros hexadecimales sin signo en la notación de ensamblador para microprocesadores de Intel.



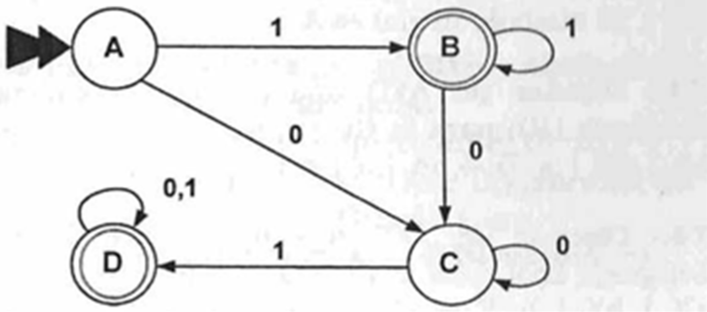
G = {(A, B, ~~C~~), (dígito, letra_hex, hache), (A→dígitoB, B→dígitoB | letra_hexB | ~~hacheC~~ | hache), (A)}

Transformado a notación en BNF:

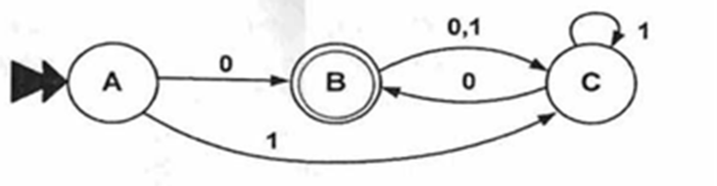
G = {(<hexadecimal>, <resto_hex>), (dígito, letra_hex, hache), (<hexadecimal>::= dígito <resto_hex>, <resto_hex>::= dígito <resto_hex> | letra_hex <resto_hex> | hache), (<hexadecimal>)}

68.- Construir una Gramática Regular que genere el Lenguaje reconocido por el Autómata Finito determinado.

a)



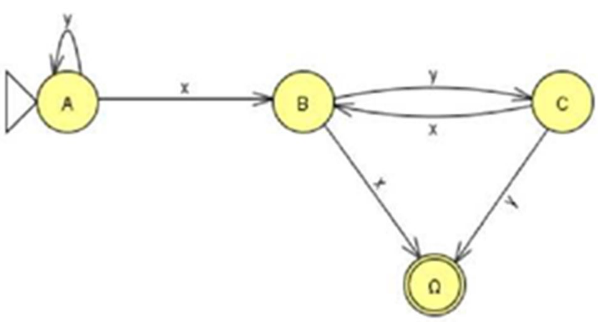
b)



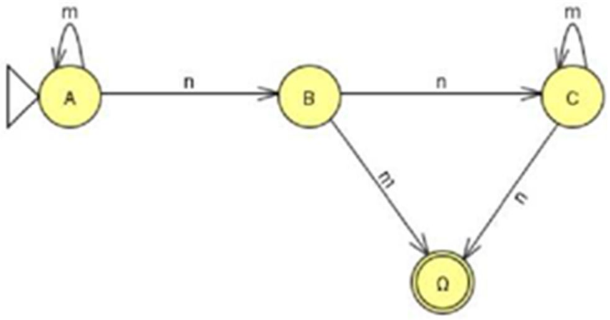
a. G = {(A, B, C, D), (0, 1), (A→1B | 0C | 1, B→1B | 0C | 1, C→1D | 0C | 1, D→1D | 0D | 0 | 1), (A)}

b. G = {(A, B, C), (0, 1), (A→0B | 1C | 0, B→0C | 1C, C→0B | 1C | 0), (A)}

69.- Diseñar el AFD que acepte las cadenas producidas por la Gramática con las composiciones P = { A → xB | yA, B → yC | x, C → xB | y } si σ_0 = { A }.

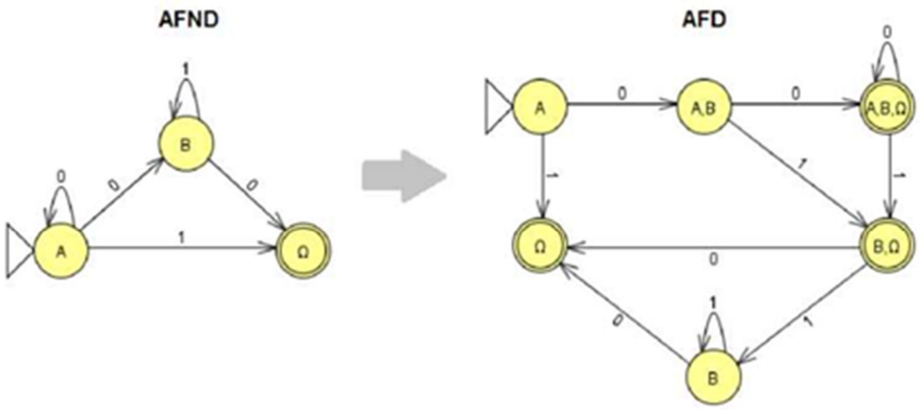


70.- Diseñar un AFD que acepte las cadenas del Lenguaje L (G) para la Gramática con reglas P = { A → mA | nB, B → nC | m, C → mC | n } con A como inicial. Caracterizar el Lenguaje producido.



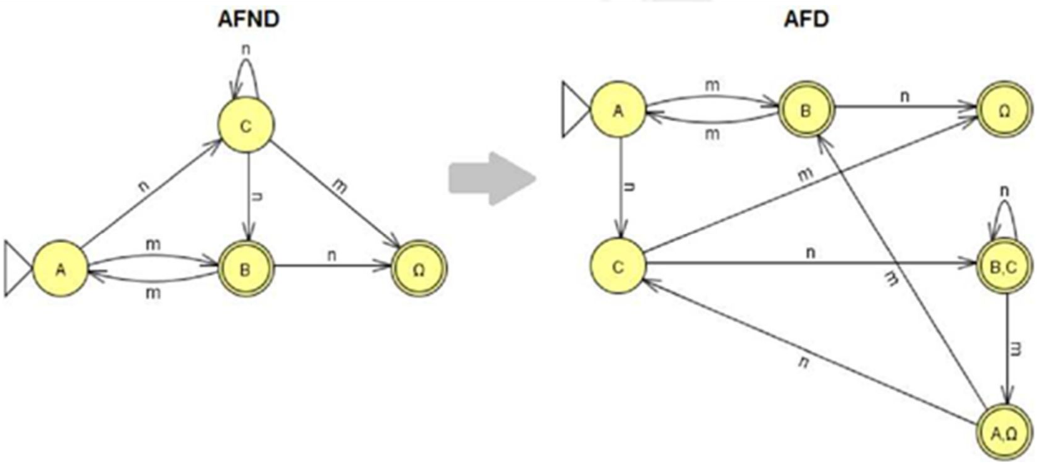
$L = \{m^*nm\} \cup \{m^*n^2m^*n\}$

71.- Diseñar un AFD que reconozca las cadenas del Lenguaje generado por la Gramática G = ({ A, B }, { 0, 1 }, { A → 0A | 0B | 1, B → 1B | 0 }, { A }). Encontrar la caracterización del Lenguaje L(G).

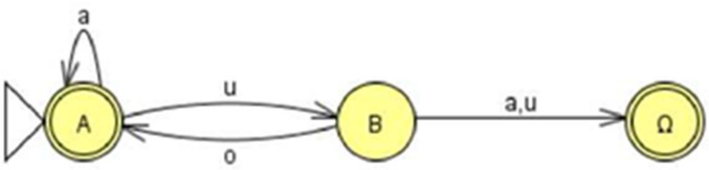


$L = \{0^*1\} \cup \{0^+1^*0\}$

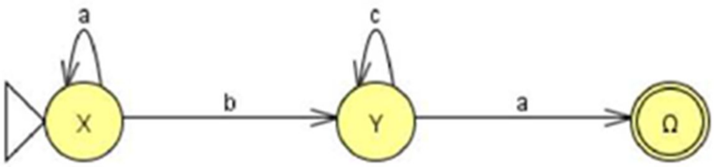
72.- Diseñar un AFD que acepte las cadenas del Lenguaje Formal L(G) para la Gramática con reglas P = { A → mB | nC | m, B → mA | n, C → nB | nC | m | n }. El símbolo inicial es A.



73.- Diseñar un AFD que acepte las cadenas del Lenguaje L(G) para la Gramática con reglas P = { A → aA | uB | a, B → oA | a | o | u } con σ_0 = { A }.

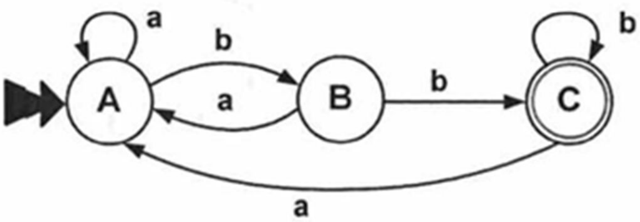


74.- Diseñar un AFD que acepte las cadenas del Lenguaje L(G) para la Gramática con reglas P = { X → aX | bY | b, Y → cY | a | c } con X como símbolo inicial. Caracterizar el Lenguaje L(G).



L = {a*bc*} ∪ {a*bc*a} = {a*bc*(λ, a)}

75.- Sea L el conjunto de arreglos aceptados por el AFD mostrado al final del enunciado. Constrúyase un AFD que acepte los arreglos L^R = { $x_n \dots x_2 x_1$ | $x_1 x_2 \dots x_n \in$ L } además de caracterizar L y L^R. Se inicia el diseño en el que era el estado de aceptación original y se traza el recorrido en sentido opuesto hasta llegar al que se tenía como el estado inicial. Finalmente se convierte el AFND en un AFD.

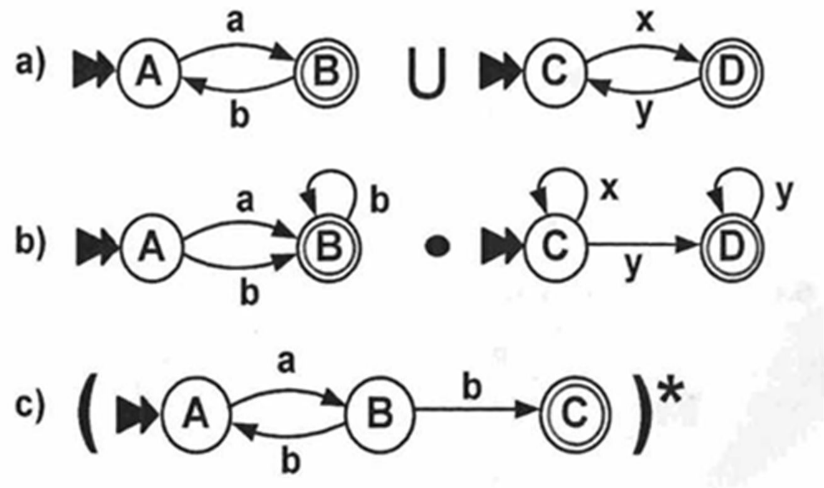


AFND **AFD**

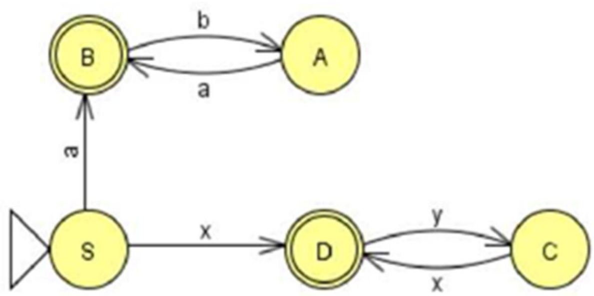
L = {(a,b)*b^2}

L^R = {b^2(a,b)*}

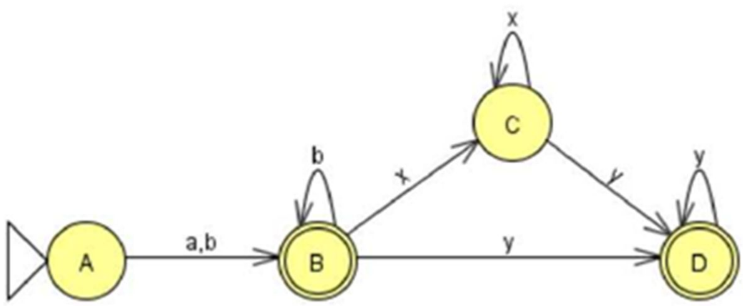
76.- Diseñar para cada inciso, el diagrama de transición del Autómata Finito que acepta las palabras, resultado de las operaciones indicadas en los correspondientes Lenguajes.



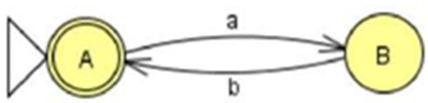
a.



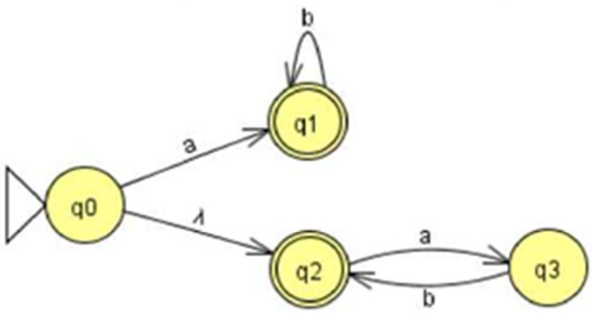
b.



c.



77.- Diseñar un Autómata Finito *con transiciones λ* en el que con I = { a, b } se acepte el Lenguaje L = {ab*} ∪ {(ab)*}. ¿Cómo sería la versión del Autómata Finito que acepta ese mismo Lenguaje pero sin *transiciones λ*?



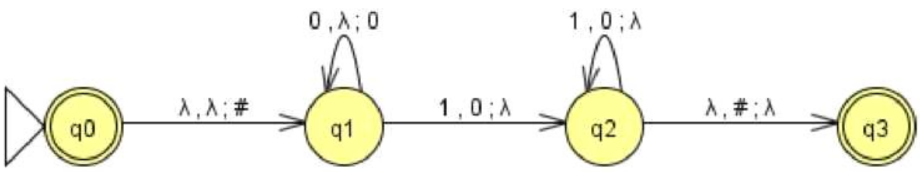
La versión del autómata finito que acepta este mismo lenguaje, L = {ab*, (ab)*}, pero sin transiciones λ, seria un automata finito no deterministico (AFND).

CAPITULO 5: Autómatas de Pila.

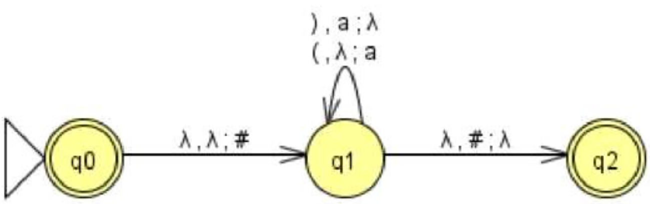
78.- Diseñar para cada inciso el Autómata de Pila que acepte el Lenguaje que se describe:

a) $L = \{ 0^n 1^n \mid n \in \mathbb{N}_0 \}$.
b) L = { s | s sea cualquier expresión aritmética que contenga paréntesis correctamente anidados }.
c) $L = \{ x^n y^n z^n \mid n \in \mathbb{N} \}$.
d) $L = \{ w^m x^n y^n z^m \mid m, n \in \mathbb{N} \}$.
e) $L = \{ w^m x^n y^m z^n \mid m, n \in \mathbb{N}_0 \}$.
f) $L = \{ x^m y^m \mid m \in \quad \} \cup \{ x^n y^{2n} \mid n \in \quad \}$.
g) L= {(a, b)* | Cantidad de a ≥ Cantidad de b }.

a.

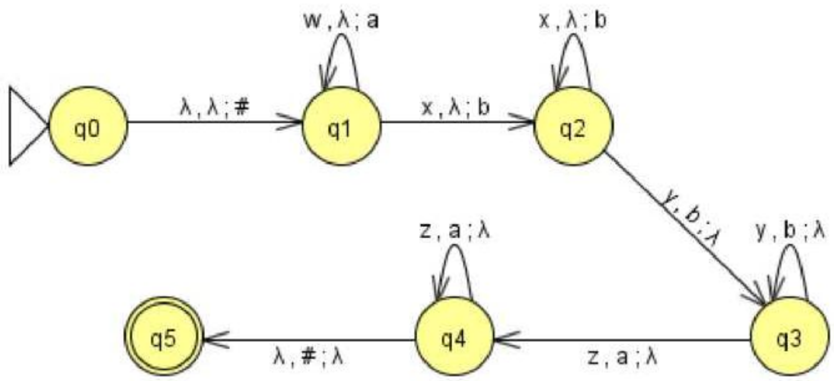


b.



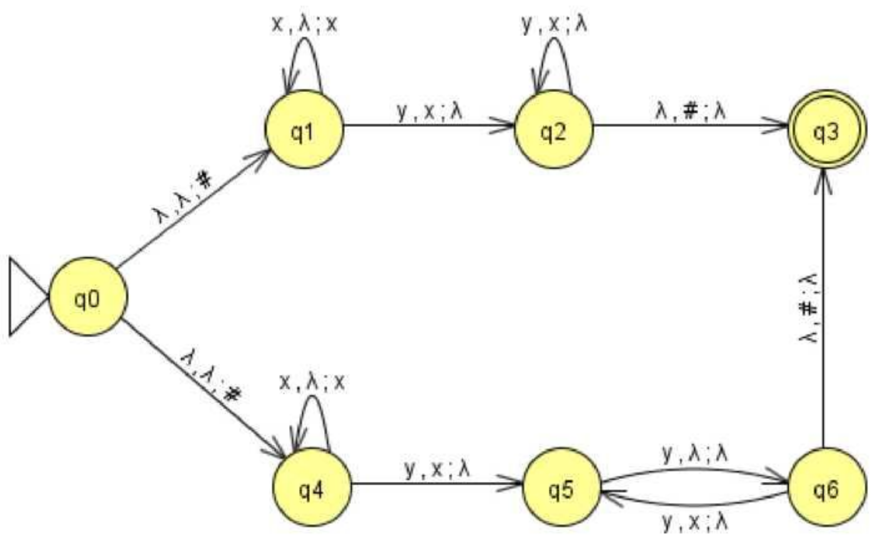
c. No existe autómata de pila que acepte este lenguaje, debido a que el lenguaje NO es libre de contexto.

d.

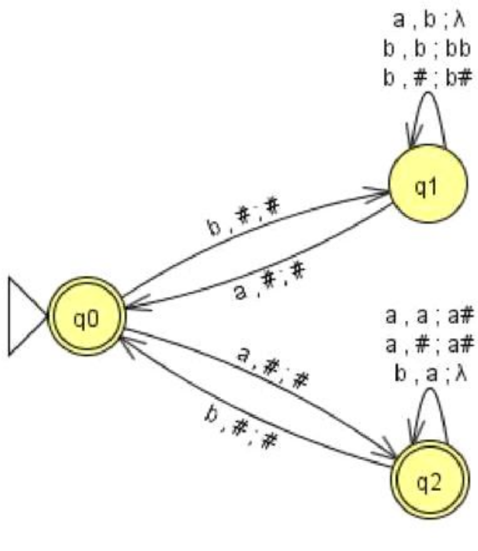


e. No existe autómata de pila que acepte este lenguaje, debido a que el lenguaje NO es libre de contexto.

f.

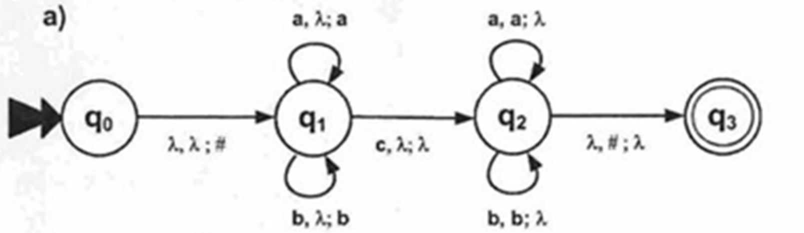


g.



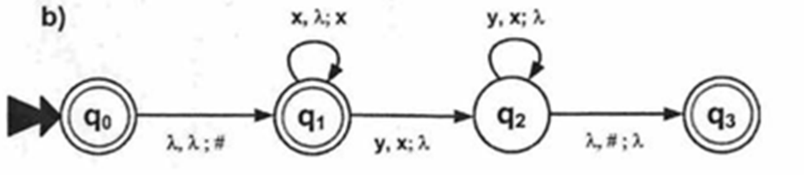
79.- ¿Qué Lenguaje Formal acepta cada de los siguientes Autómatas de Pila?

a)



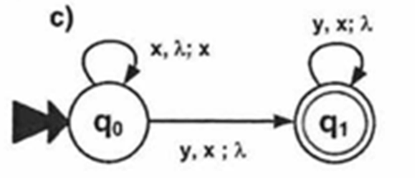
a. L = {s | s es un palíndromo de formato $(a, b)^n c(a, b)^n$}

b)



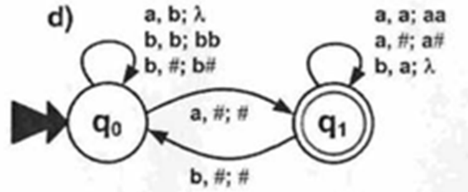
b. L = {x^m | m ∈ $\mathbb{N}_0$} ∪ {$x^n y^n$ | n ∈ ℕ}

c)



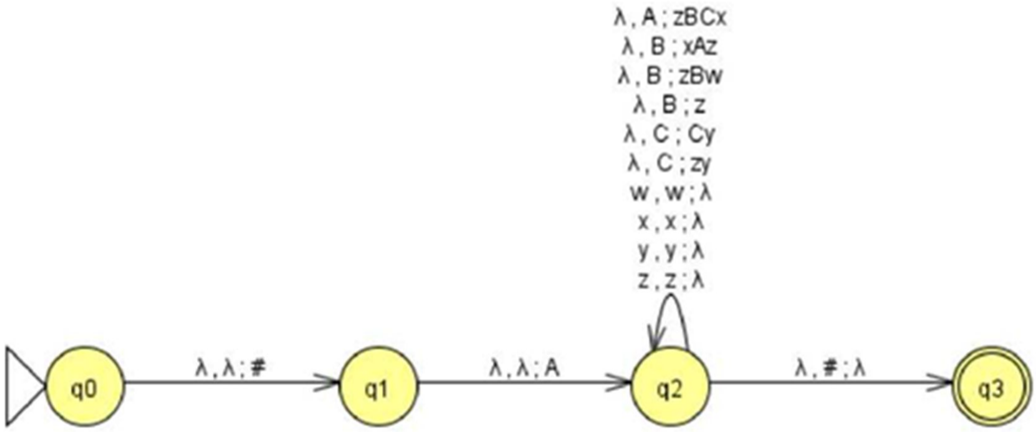
c. L = {$x^m y^n$ | m ≥ n, ∧ m, n ∈ $\mathbb{N}_0$}

d)



d. L = {$(a, b)^+$ | la cantidad de a > cantidad de b}

80.- Sea la Gramática Libre de Contexto definida por las composiciones P = { A → zBCx, B → xAz, B → zBw, B → z, C → Cy, C → zy } y símbolo inicial A. Diseñar el Autómata de Pila que acepta el Lenguaje L(G).



81.- Sea la Gramática Libre de Contexto definida por las composiciones P = { A → aBA | a, B → AbB | AA | ba } y símbolo inicial A. Diseñar el Autómata de Pila que acepta el Lenguaje L(G).