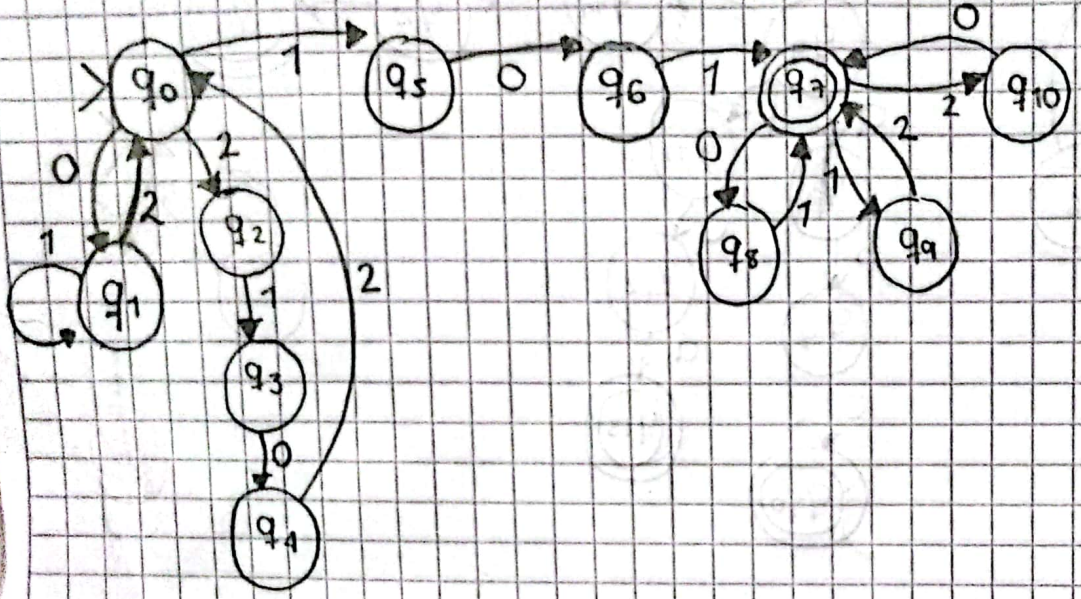
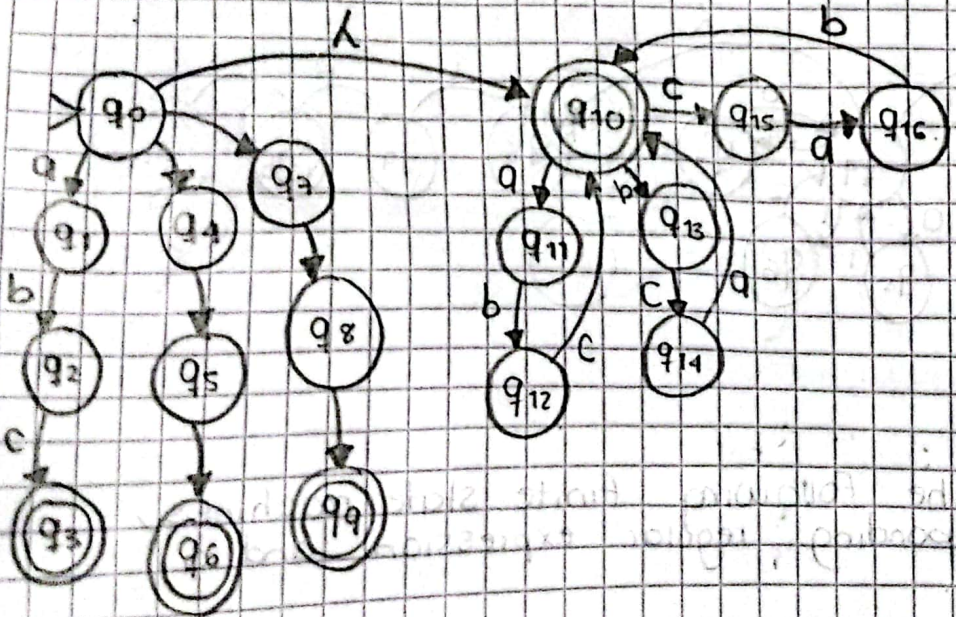
SOLUCIÓN 1.

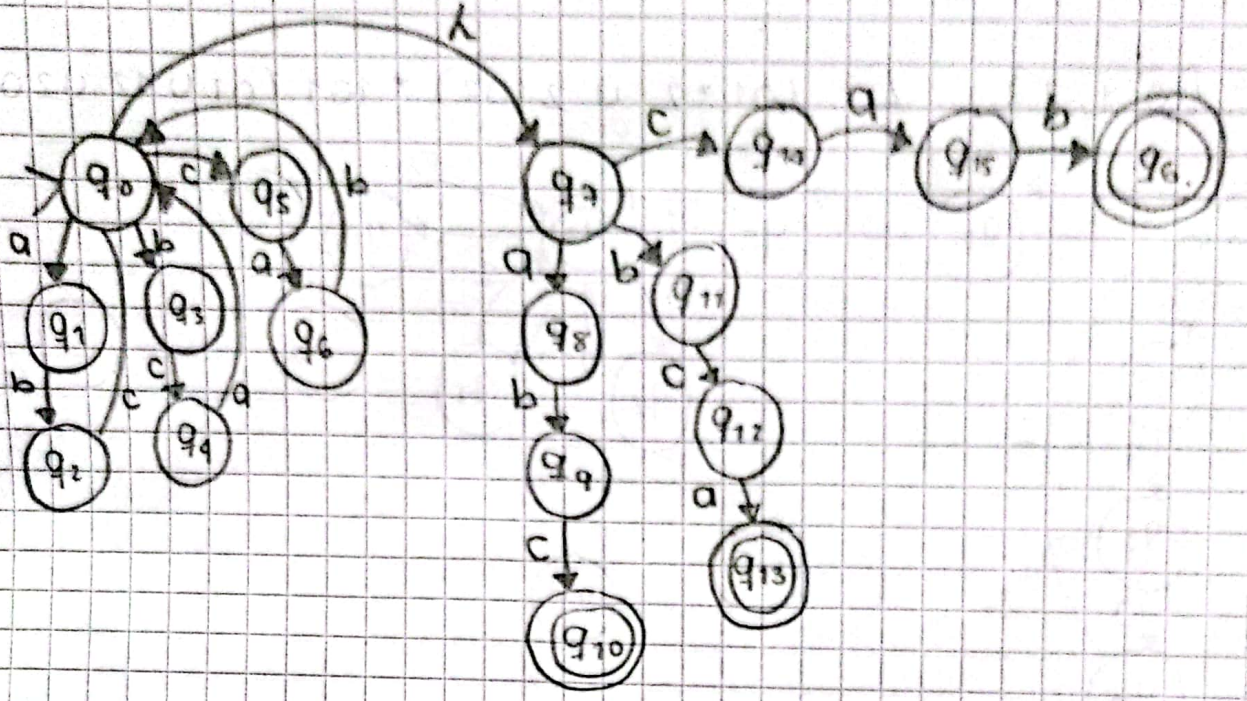
i) $\Sigma = \{0, 1, 2\}$. $L = (01^*2 \underset{Or}{\cup} 2102)^* 101 (01 \underset{Or}{\cup} 12 \cup 20)^*$



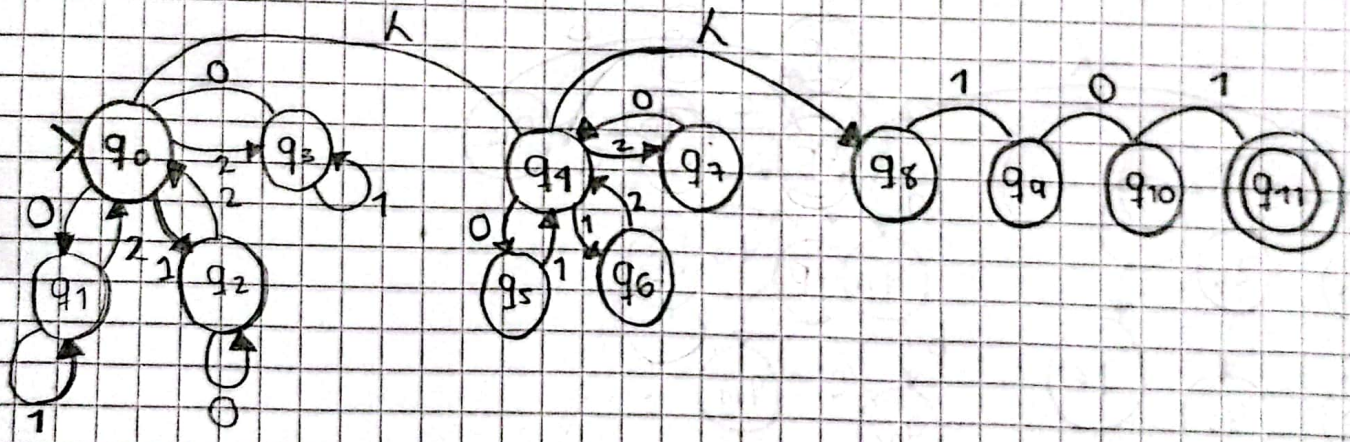
ii) $\Sigma = \{a, b, c\}$. $L = (abc \cup bca \cup cab)(abc \cup bca \cup cab)^*$



iii) $\Sigma = \{a,b,c\}$ $L = (abc \cup bca \cup cab)^* (abc \cup bca \cup cab)$



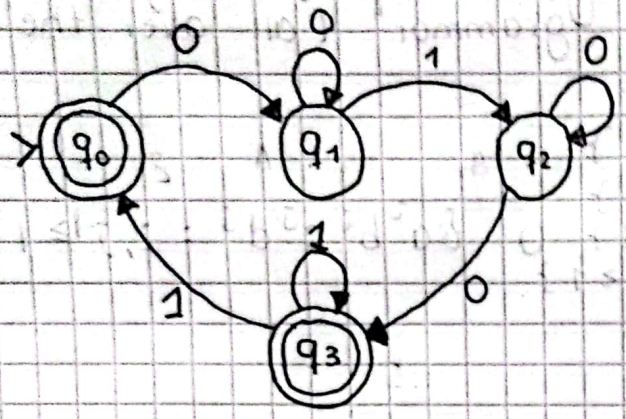
iv) $\Sigma = \{0,1,2\}$. $L = (01^*2 \cup 10^*2 \cup 21^*0)^* (01 \cup 12 \cup 20)^* 101$



2) For each one of the following finite state machines, define the corresponding regular expression and a generative grammar.

SOLUCIÓN 2.

i) $\sum = \{0, 1\}$.



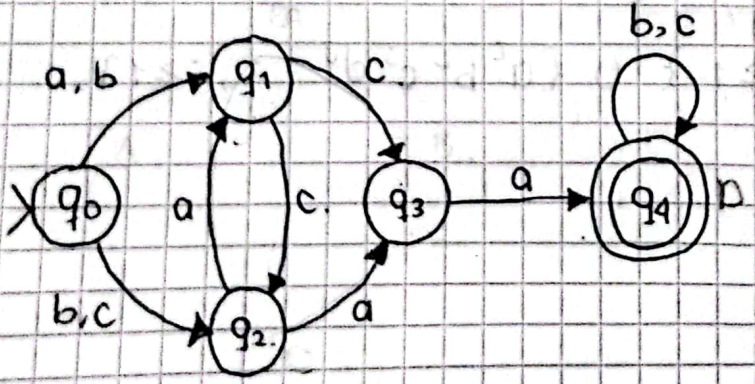
- Regular expression

$L = (00^* 10^* 01^* (\lambda \cup 1))$

Generative grammar.

$$G \begin{cases} S \to 0A \\ A \to 1B \mid 0A \\ B \to 0C \mid 0B \\ C \to 1S \mid 1C \mid \lambda \end{cases}$$

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



Generative Grammar

$S \to S_1 S_2$.

$$G \begin{cases} S_1 \to bA \mid cA \\ A \to acA \mid abD \mid acD \\ S_2 \to bC \mid cC \\ C \to caC \mid aD \mid acD \\ D \to abD \mid acD \end{cases}$$

Regular expression.

$L = (((b \cup c) \mid ac)^* a(\lambda \cup c)) \cup ((a \cup b)(ca)^* c(\lambda \cup a))a(b \cup c)^*$

3. For each of the following regular expressions, define the corresponding generative grammar (all over the alphabet $\Sigma = \{a, b, c, d\}$.

i) $\{a^i b^j c^j d^i : i, j \geq 1\}$.
ii) $\{a^i b^i c^j d^j : i, j \geq 1\}$
iii) $\{a^i b^j c^j d^i : i, j \geq 1\} \cup \{a^i b^i c^j d^j : i, j \geq 1\}$
iv) $\{a^i b^j c^{i+j} : i \geq 0, j \geq 1\}$.

SOLUCIÓN 3:

i) Terminals $\{a, b, c, d\}$.

(non terminal symbols → S, A, B.

$A = a^i, d^i$
$B = b^j, c^j$

$$G = \begin{cases} S \to AB \\ A \to aAd \mid ad \\ B \to bBc \mid bc \end{cases}$$

→ Hay un número igual de a y d y de b y c.

ii) $\{a^i b^i c^j d^j : i, j \geq 1$.

$A = a^i b^i$
$B = c^j d^j$

$$G = \begin{cases} S \to AB \\ A \to aAb \mid ab \\ B \to cBd \mid cd \end{cases}$$

iii) $\{a^i b^j c^j d^i : i, j \geq 1\} \cup \{a^i b^i c^j d^j : i, j \geq 1\}$.

S → Start symbol.

$S \to S_1 \mid S_2$.
$S_1 \to AB$
$A \to aAd \mid ad$
$B \to bBc \mid bc$
$S_2 \to CD$.
$C \to aAb \mid ab$
$D \to cBd \mid cd$.

iv) $\{a^i b^j c^{i+j} : i \geq 0, j \geq 1\}$.

$S \to ABC$.
$A \to aA \mid \lambda$. → Genera la cantidad de a las veces $i \geq 0$.
$B \to bB \mid b$
$C \to cC \mid c$.

4. Be G a context - free grammar with the following productions =

$$G = \begin{cases} S \to ABC \mid BaC \mid aB. \\ A \to Aa \mid a. \\ B \to BAB \mid bab. \\ C \to cC \mid \lambda. \end{cases}$$

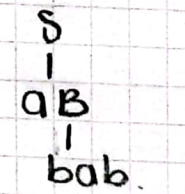
Found derivation trees for the following strings =

i) w_1 = abab.
ii) w_2 = babacc.
iii) w_3 = ababababc.

SOLUTION 4.

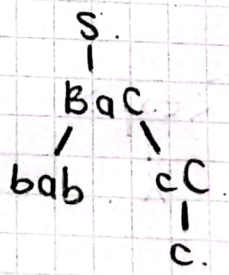
i) w_1 = abab. Derivation tree.

$S \to aB.$
$B \to bab.$



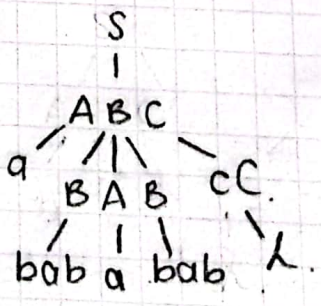
ii w_2 = babacc.

$S \to BaC.$
$B \to bab.$
$C \to cC \to cc.$



iii) w_3 = ababababc.

$S \to ABC.$
$A \to Aa \to aAa \to aa.$
$B \to bab.$
$C \to c.$



5 As follows there is a context-free grammar to generate real numbers without sign, the alphabet is

$\Sigma = \{0,1,2,3,4,5,6,7,8,9,.,+,-,E\}$.

<real> → <digits> <decimal> <exp>.
<digits> → <digits> <digits> | 0|1|1|2|3|4|5|6|7|8|9.
<decimal> → <digits> | λ.
<exp> → E <digits> | E+ <digits> | E- <digits> | λ.

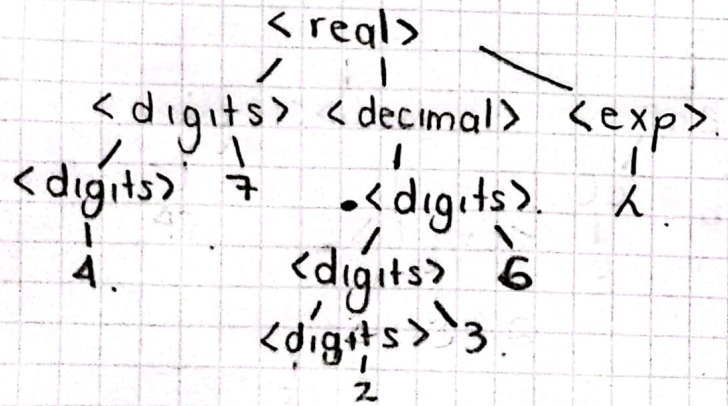
Define the derivation tree for the following strings.

i) w_1 = 47,236.
ii) w_2 = 321,25 E+35.
iii) w_3 = 0,8 Eq.
iv) w_4 = 0,8 E+9.

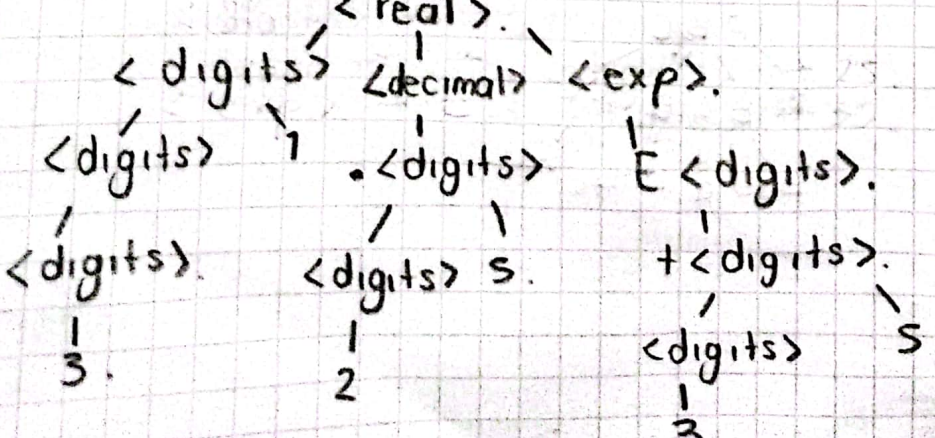
SOLUTION 5.

i) w_1 = 47,236.

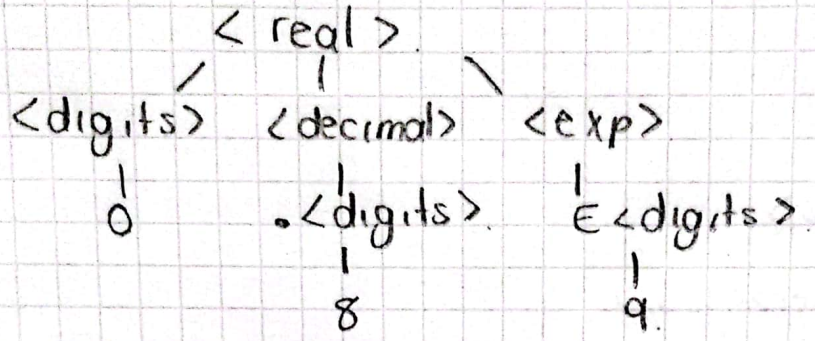
<digits> → 47.
<decimal> → 236.
<exp> → λ.



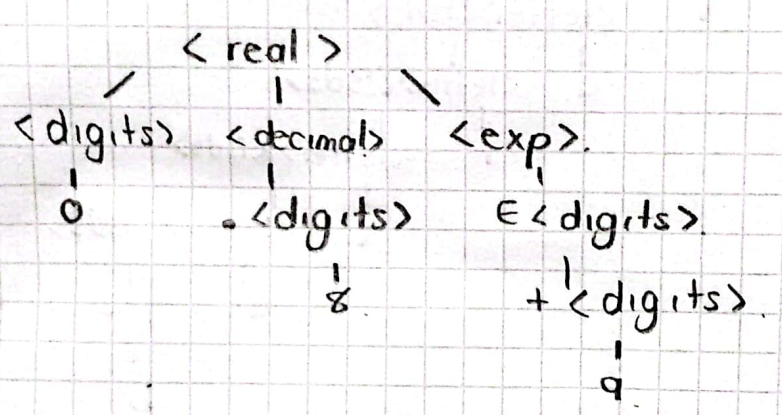
ii).



iii) $w_3 = 0,8 \in 9$.



iv) $w_4 = 0,8 \in +9$.



6. As follows there is a context free grammar to generate identifiers, identifiers are strings of letters and digits, starting with a letter.

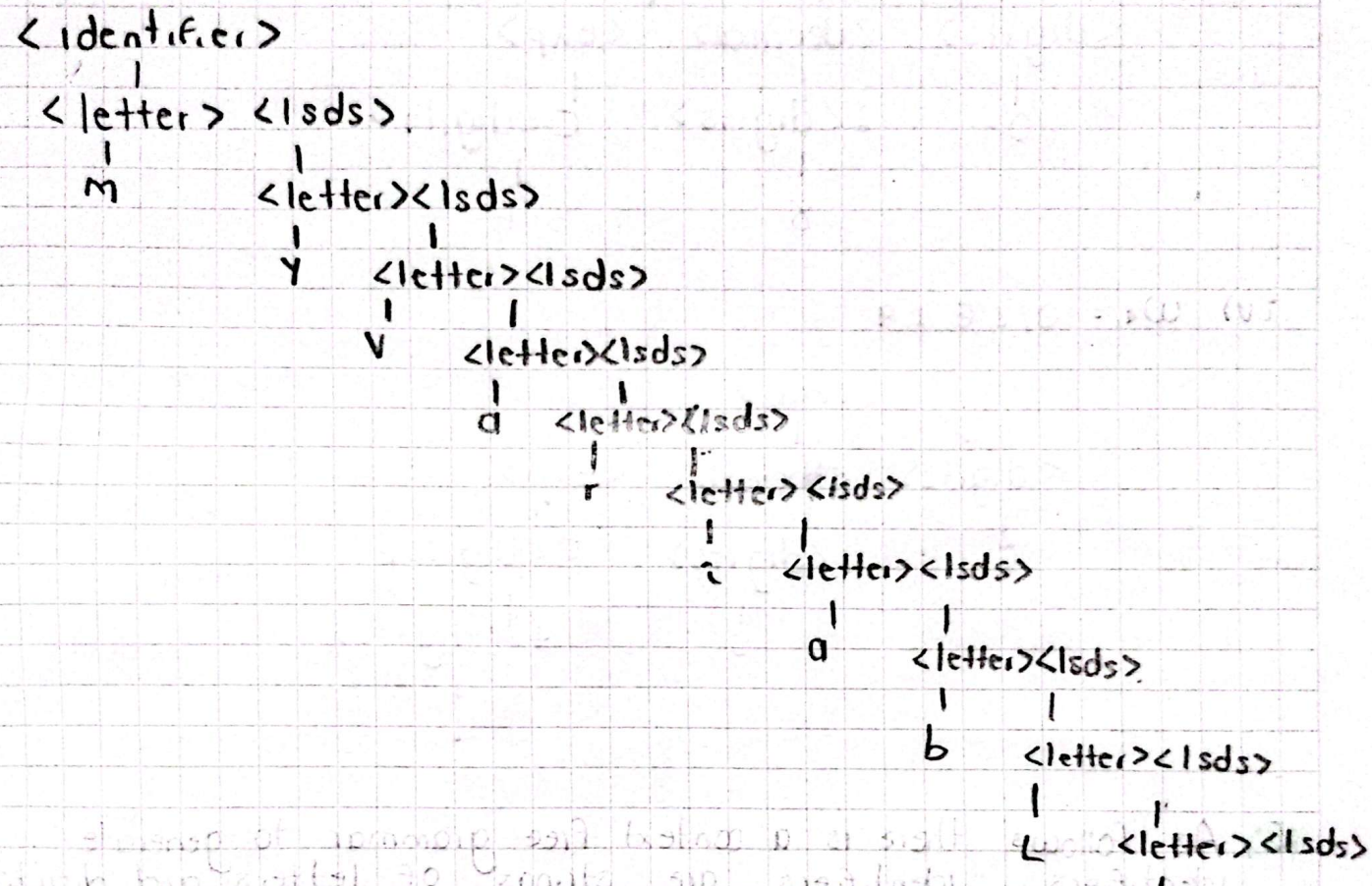
<identifier> → <letter><lsds>.
<lsds> → <letter><lsds> | <digit><lsds> | λ.
<letter> → a|b|c|...|x|y|z|A|B|C|...|X|Y|Z.
<digit> → 0|1|2|3|4|5|6|7|8|9.

Define the derivation tree for the following names.

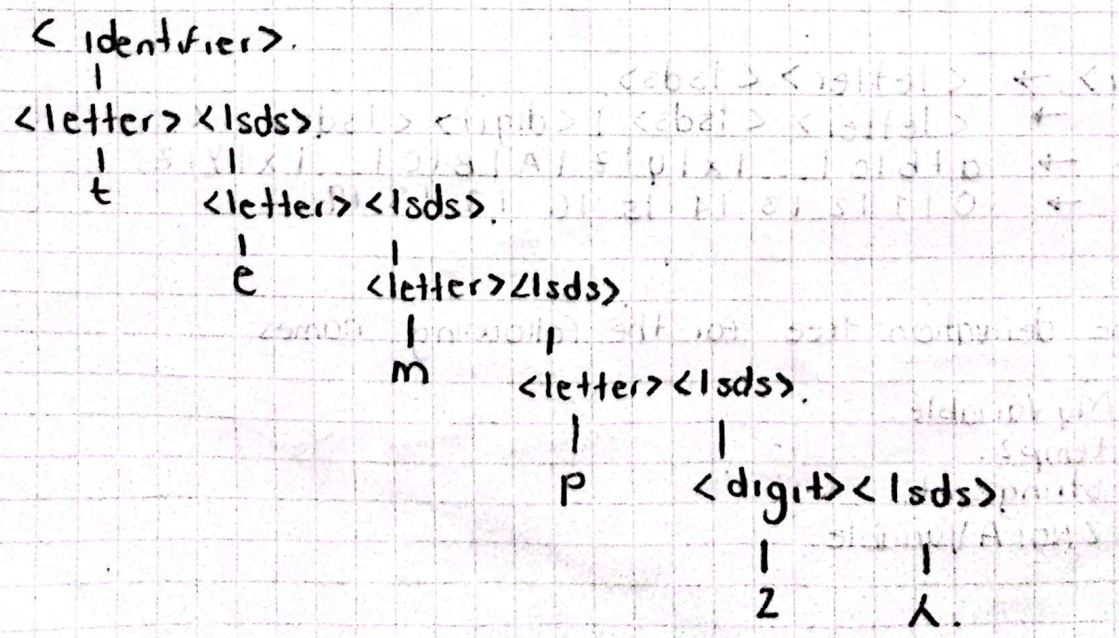
i) w_1 = My Variable.
ii) w_2 = temp2.
iii) w_3 = string2int.
iv) w_4 = 2NotAVariable.

SOLUTION 6.

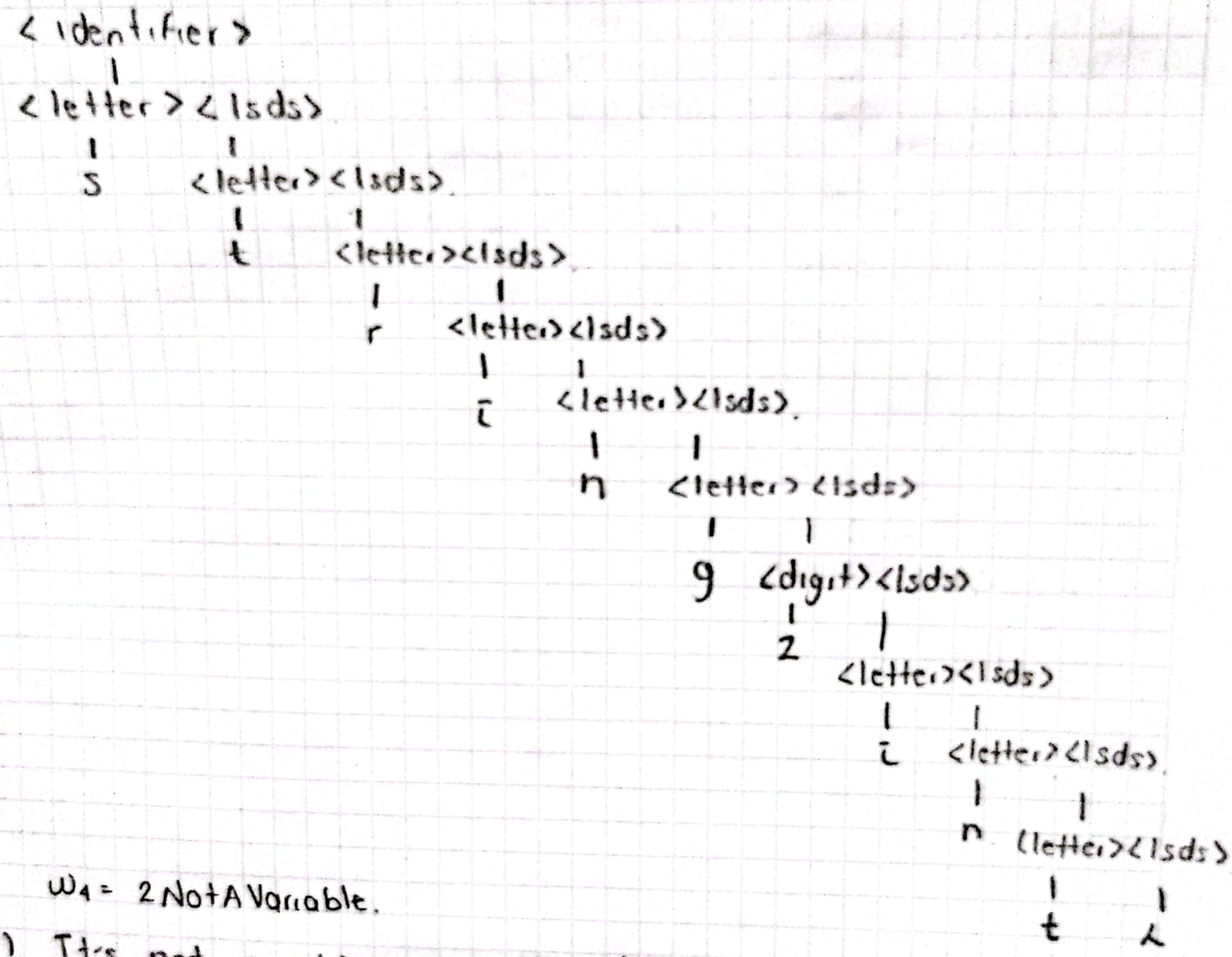
i) W_1 = MyVariable.



ii). W_2 = temp2.



iii). w_3 = string 2 int.



w_4 = 2NotAVariable.

iv). It's not possible make the derivation tree due to the identifier start with a digit and the grammar specifies that identifiers must start with a letter.

<identifier> → <letter> <lsds>