**Práctico 4 - Gramáticas libres de contexto**

**Ejercicio 1**

Para cada uno de los siguientes lenguajes definidos sobre el alfabeto A = {a, b, c, d, e, h, x, y, z, 0, 1, 2, 3, 4} diseñar y definir formalmente una GLC que lo reconozca:

1. L1 = {a²k b2n ck dj / k, n, j >= 0}

|  |  |  |  |
| --- | --- | --- | --- |
| k | n | j | Cadena resultante |
| 0 | 0 | 0 | λ |
| 0 | 0 | 1 | d |
| 0 | 1 | 0 | bb |
| 1 | 0 | 0 | aac |
| 0 | 1 | 1 | bbd |
| 1 | 1 | 1 | aabbcd |

Base = aak bbn ck dj

S => λ | C | A | AC

----------------------------

A => aa A c | aac | B

B => bbB | bb

C => dC | d

1. L2 = {xr ys zt / t = r+s, r, s >= 1}

Posibles: xyzz, xxyyzzzz

Base = xr ys zr zs

S => xAz

-----------------

A => xAz | B

B => yBz | yz

1. L3 = {xr ys zt / s = r+t, r,s >= 1}

Despeje del exponente s = r + t <==> 1 = 1 + t <==> 1 - 1 = t <==> t = 0 ====> t >= 0.

Posibles:

|  |  |  |  |
| --- | --- | --- | --- |
| r | s | t | Cadena resultante |
| 0 | 0 | 0 | λ |
| 1 | 1 | 0 | xy |
| 0 | 1 | 1 | yz |
| 1 | 2 | 1 | xyyz |

Base = xr yryt zt

S => λ | A | B | AB

--------------------------

A => xAy | xy

B => yBz | yz

1. L4 = { x / x = a Y e, donde Y = b3n c d3n, n >= 1}.

Posibles: abbbcddde, abbbbbbcdddddde.

Base: a (bbb)n c (ddd)n e

S => abbbAddde

----------------------------

A => bbb A ddd | c

1. L5 = {1n 0k / n >= 0 y k = 3n}.

Posibles = 1000, 11000000, 111000000000.

Base = 1n (000)n

S => 1 S 000 | 1000

1. L6 = { (ab)j c2i bi+1 ck dn  / i, k, n >= 0 y n < j}.

Posibles:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| j | i | k | n | Cadena resultante |
| 1 | 0 | 0 | 0 | abb |
| 1 | 0 | 0 | 1 | abbd |
| 1 | 0 | 1 | 0 | abbc |
| 1 | 0 | 1 | 1 | abbcd |
| 1 | 1 | 0 | 0 | abccbb |
| 1 | 1 | 0 | 1 | abccbbd |
| 1 | 1 | 1 | 0 | abccbbc |
| 1 | 1 | 1 | 1 | abccbbcd |

Base = (ab)j (cc)i bi **b** ck dn

S => Ab | AbD | AbC | AbCD

---------------------------------------------------

A => abA | ab | B

B => ccBb | ccb

C => cC | c

D => dD | d

1. L7 = { an bn+2 am ek bm+1 / n, m >= 1 y k >= 0}.

Posibles:

|  |  |  |  |
| --- | --- | --- | --- |
| n | m | k | Cadena resultante |
| 1 | 1 | 0 | ab**bb**ab**b** |
| 1 | 1 | 1 | ab**bb**aeb**b** |

Base = an bn **bb** am ek bm **b**

S => AbbBb

----------------------------------

A => aAb | ab

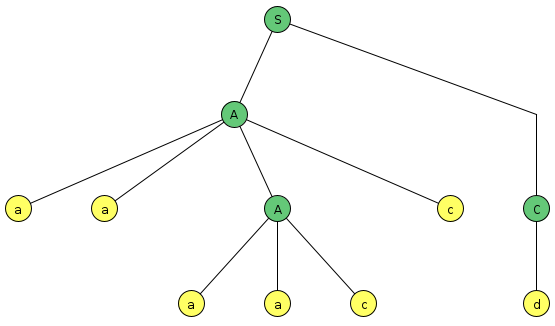
B => aBb | aCb | ab

C => eC | e

**Ejercicio 2**

Para cada una de las siguientes hileras armar el árbol de derivación en cada una de las gramáticas definidas en el ejercicio anterior.

1. h1 = aaaacccd.

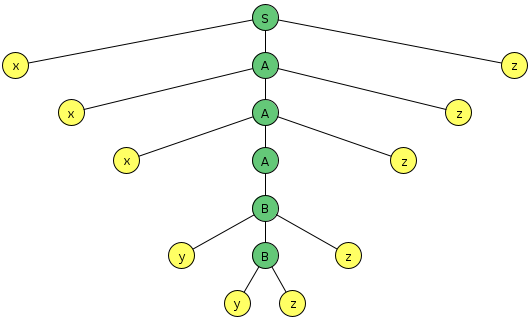


h1 = aaaacccd => No pertenece a la G definida y por lo tanto no se puede hacer su árbol de derivación.

Resultado más cercano = aaaaccd.

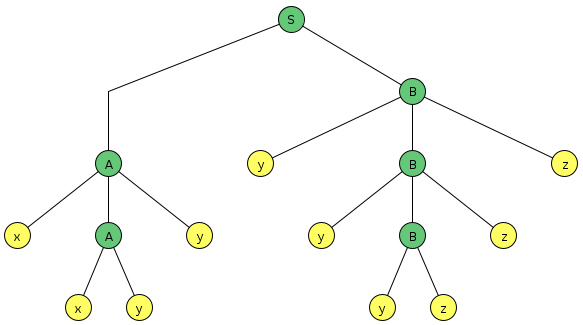
Sobra una “c” en h1.

1. h2 = xxxyyzzzzz.

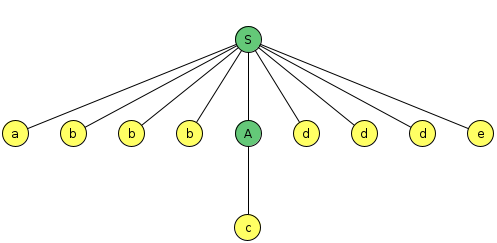


La cadena pertenece a la G definida.

1. h3 = xxyyyyyzzz.

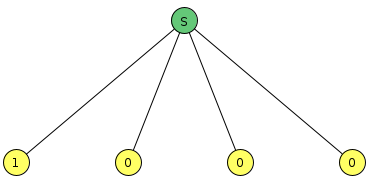


La cadena pertenece a la G definida.

1. h4 = abbbcddde.

La cadena pertenece a la G definida.

1. h5 = 11000.

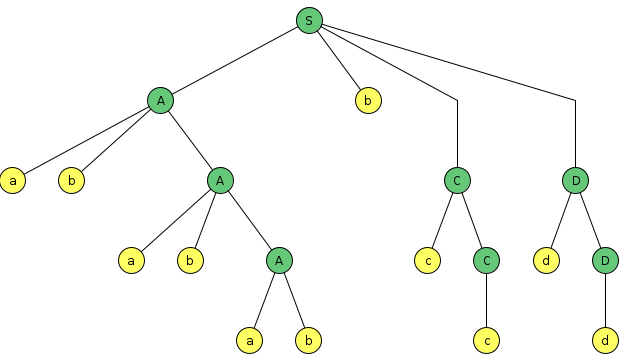


h5 = 11000 => No pertenece a la G definida y por lo tanto no se puede hacer su árbol de derivación.

Resultado más cercano = 1000 .

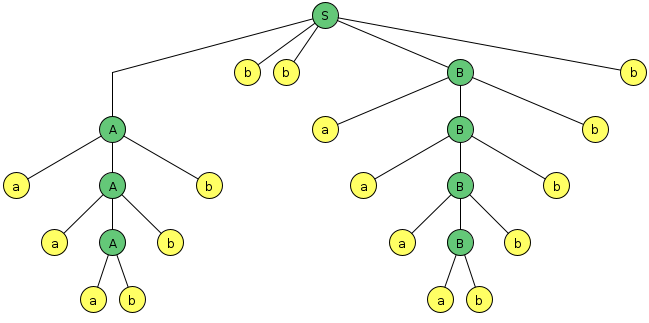
Sobra un “1” en h5.

1. h6 = abababbccdd.



La cadena pertenece a la G definida.

1. h7 = aaabbbbbaaaabbbbb.



La cadena pertenece a la G definida.

**Ejercicio 3:**

Dada la siguiente gramática ambigua:

A => 0BB

B => 1A | 0A | 0

1. Mostrar los diferentes árboles de derivación para una hilera adecuada.
2. Clasificar el lenguaje que escribe esta gramática.

**Ejercicio 4:**

**Ejercicio 6:**

<cabecera\_Pro> ::= procedure <nombre> ( <list\_param> )

<Nombre> ::= <letra> / <nombre> <letra> / <nombre> <digito>

<letra> ::= a/b/c/.../z

<digito> ::= 0/1/2/.../9

<numero> ::= <digito> / <numero> <digito>

<list\_param> ::= <param> / <list\_param> , <param>

<param> ::= <nombre> <tipo> <EntSal> / <nombre> <tipo> <EntSal> default <Default\_type>

<tipo> ::= char / number / string / bool

<EntSal> ::= IN / OUT /INOUT

<Default\_type> ::= <nombre> / <numero>