Pumping Lemma

Lemma : *The pumping lemma states that for a regular language L, there exists a constant n such that every string w in L (such that length of w (|w|) >= n) we can break w into three substrings, w = xyz, such that*

- y is not null or Y!=€
- |xy| <= n
- For all i >= 0, the string xy^iz is also in L

Note that |x| and |z| can be 0 but |y| has to be greater than 1.

The lemma states that for a regular language every string can be broken into three parts x,y and z such that if we repeat y i times between x and z then the resulting string is also present in L.

1. Prove that Language L = {a^nb^n for n>=0} is not regular.

L={€,ab,aabb,aaabbb,aaaabbbb,aaaaabbbbb,..........}

w=ab = xyz

x=a y=b z=€

w= xy^iz= ab^i€

i=2 abb not belongs to L

i=3 abbb not belongs to L
hence the given language L is not regular.

2. Prove that Language L = { 0^n10^n n>=1} is not regular.

L={010,00100,0001000,000010000,.........}

w= 0001000 =xyz

$x=00$; $y=01$; $z=000$

$w = xy^iz = 00\ (01)^i\ 000$

consider i= 2 000101000 not belongs to L

Hence ,the given language is not regular.

3. Prove that Language $L = \{ 0^n1^{2n} \ \ n>=1\}$ is not regular.

L={ 011,001111,000111111,000011111111,........}

w=001111 =xyz

x=00 y=11 z=11

$xy^iz = 00\ (11)^i\ 11$

consider i=2 00111111 not belongs to L

Hence ,the given language is not regular.

Context Free Grammars (CFG)

CFG G= (V, T, P, S)
V= Set of variables (or) Non-Terminals→ All Capital letters in a grammar
T= Set of Terminals→ All small letters, numbers, special symbols
P= Set of Productions → Ex: A→Bc , Variable→ Body of the production
S= Start Symbol of the grammar

Consider A: 1)P→€ 2)P→0 3)P→1 4)P→0P0 5)P→1P1

CFG G=({P}, {0,1}, A, P)

Derivation:
Consider the string a)10101 (or) b)101101
a)
P→1P1
→1 0P0 1
→1 0 1 0 1

b) P→1P1
→ 1 0P0 1
→ 1 0 1P1 0 1
→ 101€101
→ 101101

Derivation:
1. Left-most Derivation
2. Right-most Derivation

Top-Down Approach→ Head or Start symbol (Root) to Body {Expansion}

Bottom-Up Approach → Body to Head (Root) or Start symbol {Reduction}

Consider the following grammar

1.	E	$\rightarrow$	I
2.	E	$\rightarrow$	$E + E$
3.	E	$\rightarrow$	$E * E$
4.	E	$\rightarrow$	(E)
5.	I	$\rightarrow$	a
6.	I	$\rightarrow$	b
7.	I	$\rightarrow$	Ia
8.	I	$\rightarrow$	Ib
9.	I	$\rightarrow$	$I0$
10.	I	$\rightarrow$	$I1$

Using the above grammar derives the string: a*(a+b00)
CFG G=({E,I}, {0,1,a,b,+,*,(,)}, A, E)

Left-most Derivation: for a*(a+b00)	Right-most Derivation: for a*(a+b00)
E → E*E	**E** → E*E
→ I*E	→ E*(E)
→ a*E	→ E*(E+E)
→ a*(E)	→ E*(E+I)
→ a*(E+E)	→ E*(E+I0)
→ a*(I+E)	→ E*(E+I00)
→ a*(a+E)	→ E*(E+b00)
→ a*(a+I)	→ E*(I+b00)
→ a*(a+I0)	→ E*(a+b00)
→ a*(a+I00)	→ I*(a+b00)
→ a*(a+b00)	→ a*(a+b00)
Stops here	Stops here

$$
\begin{aligned}
S &\rightarrow A1B \\
A &\rightarrow 0A \mid \epsilon \\
B &\rightarrow 0B \mid 1B \mid \epsilon
\end{aligned}
$$

Give leftmost and rightmost derivations of the following strings:

* a) 00101.

b) 1001.

c) 00011.

LMD:

String : 00101

S →A1B
→ 0A1B
→00A1B
→00€1B
→0010B
→00101B
→00101€
→00101

RMD:

S →A1B
→A10B
→A101B
→A101€
→0A101
→00A101
→00€101
→00101

Ambiguous Grammars:

CFG G=(V,T,P,S)

Consider the grammar

E→ E+E | E*E | I

I→ 0|1|a|b

Derive a string : a+b*a

Case i: LMD

E → E+E
→I+E
→a+E
→a+E*E
→a+I*E
→a+b*E
→a+b*I
→a+b*a

Case ii: LMD

E → E*E
→ E+E*E
→I+E*E
→a+E*E
→a+I*E
→a+b*E
→a+b*I
→a+b*a

If the grammar is ambiguous, it generates two LMDs, or two RMDs or two parse trees for the string.

Consider the grammar

E→ E+E   E→E*E   E→(E)   E→id

String: id+id*id

E → E+E
→id+E
→id+E*E
→id+id*E
→id+id+id

E →E*E
→E+E*E
→id+E*E
→id+id*E
→id+id*id

LMD:

String : 00101

S →A1B
→ 0A1B
→00A1B
→00€1B
→0010B
→00101B
→00101€
→00101

RMD:
S →A1B
→A10B
→A101B
→A101€
→0A101
→00A101
→00€101
→00101

→00101
→00€101
→00A101
→0A101
→A101€
→A101B
→A10B
→A1B
→S

Push Down Automata

Example Problems:

1. Design a PDA for L={ww^r | w € (0+1)*}

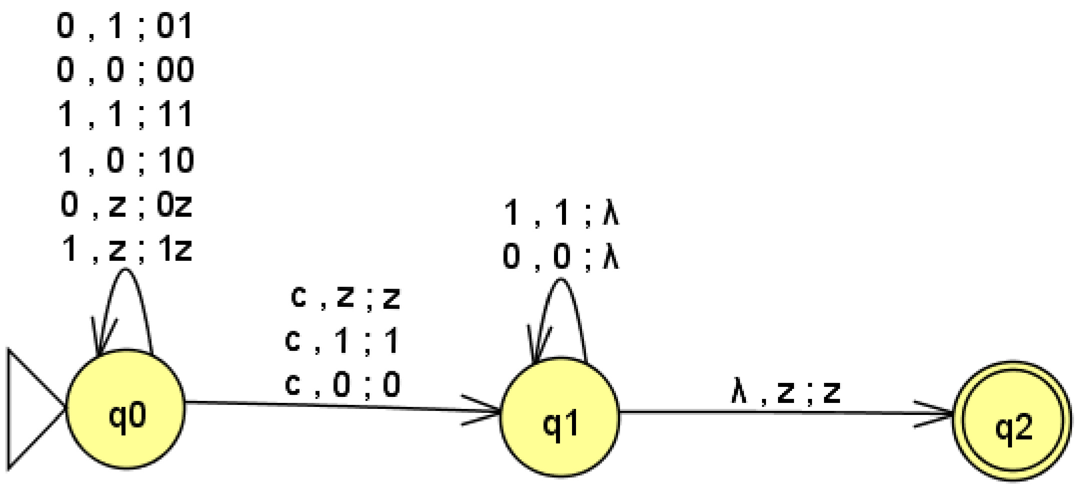
w=100 w^r = 001

L1= w€w^r = 100€001 → even Palindrome

2. Design a PDA for L={wcw^r | w € (0+1)*}

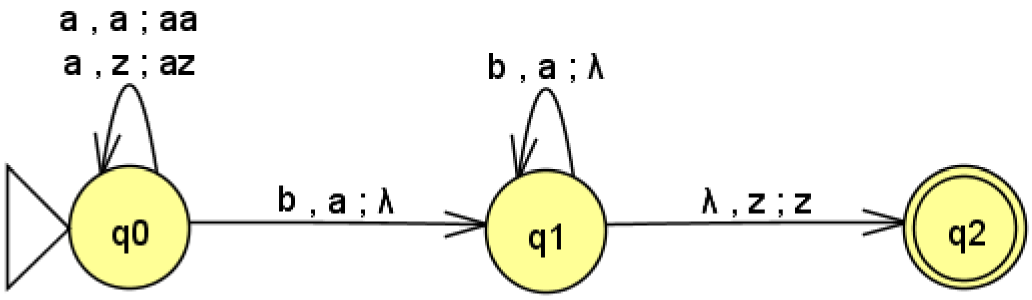
L2= wcw^r =100c001 → odd Palindrome

= 01c10



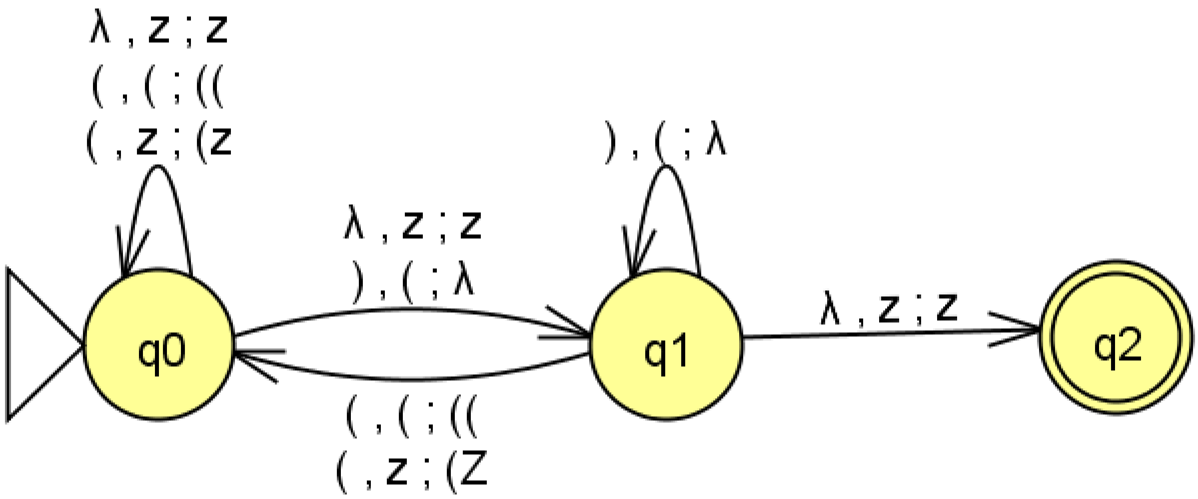
3. Design a PDA for L={ a^n b^n | n >= 1 }

L= { ab, aabb, aaabbb, aaaabbbb,aaaaabbbbb,.........}



4. Design a PDA for L={ balanced parenthesis }

L= { ( ), (( )), (( )( )), ...}



Equivalence of PDAs and CFGs

The above informal construction can be made precise as follows. Let $G = (V, T, Q, S)$ be a CFG. Construct the PDA P that accepts $L(G)$ by empty stack as follows:

$$P = (\{q\}, T, V \cup T, \delta, q, S)$$

where transition function δ is defined by:

1. For each variable A,

$$\delta(q, \epsilon, A) = \{(q, \beta) \mid A \to \beta \text{ is a production of } G\}$$

2. For each terminal a, $\delta(q, a, a) = \{(q, \epsilon)\}$.

Example 6.12: Let us convert the expression grammar of Fig. 5.2 to a PDA. Recall this grammar is:

$$\begin{array}{lcl} I & \to & a \mid b \mid Ia \mid Ib \mid I0 \mid I1 \\ E & \to & I \mid E * E \mid E + E \mid (E) \end{array}$$

1.Variables in G: I, E

Terminals in G: a, b, 0, 1, *, +, (,)

PDA P= ({q}, {a,b,0,1,*,+,(,) }, V U T, δ,q,I)

From Rule 1:

δ(q,€,I) = { (q,a) }

δ(q,€,I) = { (q,b) }

δ(q,€,I) = { (q,Ia) }

δ(q,€,I) = { (q,Ib) }
δ(q,€,I) = { (q,I0) }
δ(q,€,I) = { (q,I1) }

δ(q,€,E) = { (q,I) }
δ(q,€,E) = { (q,E*E) }
δ(q,€,E) = { (q,E+E) }
δ(q,€,E) = { (q,(E)) }

From Rule 2:

δ(q,a,a) = { (q,€) }
δ(q,b,b) = { (q,€) }
δ(q,0,0) = { (q,€) }
δ(q,1,1) = { (q,€) }
δ(q,*,*) = { (q,€) }
δ(q,+,+) = { (q,€) }
δ(q,(,( ) = { (q,€) }
δ(q,),)) = { (q,€) }

* **Exercise 6.3.1:** Convert the grammar

$$\begin{array}{lcl} S & \rightarrow & 0S1 \mid A \\ A & \rightarrow & 1A0 \mid S \mid \epsilon \end{array}$$

to a PDA that accepts the same language by empty stack.

Exercise 6.3.2: Convert the grammar

$$\begin{array}{lcl} S & \rightarrow & aAA \\ A & \rightarrow & aS \mid bS \mid a \end{array}$$

to a PDA that accepts the same language by empty stack.

2.
Variable in G: S, A
Terminals in G : 0, 1.
PDA A=({q}, {0,1}, V U T, δ, q, S)
From rule 1:
δ(q, €, S) = { (q, 0S1) }
δ(q, €, S) = { (q, A) }

$\delta(q, €, A) = \{ (q, 1A0) \}$
$\delta(q, €, A) = \{ (q, S) \}$
$\delta(q, €, A) = \{ (q, €) \}$
From rule 2:

$\delta(q, 0,0) = \{ (q, €) \}$

$\delta(q, 1,1) = \{ (q, €) \}$

3.
Variables in G: S, A
Terminals in G: a, b
PDA A= ({q}, {a,b}, V U T, δ, q, S)

Rule 1:

$\delta(q, €, S) = \{ (q, aAA) \}$

$\delta(q, €, A) = \{ (q, aS) \}$

$\delta(q, €, A) = \{ (q, bS) \}$

$\delta(q, €, A) = \{ (q, a) \}$

Rule 2:

$\delta(q, a, a) = \{ (q, €) \}$

$\delta(q, b, b) = \{ (q, €) \}$

Normal Forms for CFG

1. CNF (Chomsky Normal Form)
2. GNF (Greibach Normal Form)

Input: Context Free Grammar (CFG)

if any,

1. Eliminate €-Production from the given CFG (Ex: A→€)
2. Eliminate Unit Productions from the given CFG (Ex: A→B)
3. Eliminate Useless Symbols from the given CFG (Not generated anything)

Output: Either CNF or GNF → Context Free Grammar

1. Elimination of €-Productions:

Consider the grammar

$$
\begin{aligned}
S &\to AB \\
A &\to aAA \mid \epsilon \\
B &\to bBB \mid \epsilon
\end{aligned}
$$

There are two €-Productions : A→€ and B→ €

S→ AB | A | B
A→ aAA | aA | aA | a

B→ bBB | bB | bB | b

Now the resultant grammar after elimination of €-Productions

S→ AB | A | B
A→ aAA | aA | a
B→ bBB | bB | b

Unit Productions are: S→ A and S→ B

Resultant grammar after elimination of unit productions:

S→ ABC | aAA | aA | a | bBB | bB | b
A→aAA | aA | a
B→ bBB | bB | b

The grammar has useless symbol: C

S→ aAA | aA | a | bBB | bB | b
A→aAA | aA | a
B→ bBB | bB | b

Example for elimination of unit Productions

Consider the grammar

$$
\begin{aligned}
I &\rightarrow a \mid b \mid Ia \mid Ib \mid I0 \mid I1 \\
F &\rightarrow I \mid (E) \\
T &\rightarrow F \mid T * F \\
E &\rightarrow T \mid E + T
\end{aligned}
$$

There are three unit productions (A→ B) in the given CFG
F→ I   T→F   E→T

F→ a | b | Ia | Ib | I0 |I1 | (E)

T→ a | b | Ia | Ib | I0 |I1 | (E) | T*F

E→ a | b | Ia | Ib | I0 |I1 | (E) | T*F | E+T

The resultant grammar is:

I→ a | b | Ia | Ib | I0 |I1

F→ a | b | Ia | Ib | I0 |I1 | (E)

T→ a | b | Ia | Ib | I0 |I1 | (E) | T*F

E→ a | b | Ia | Ib | I0 |I1 | (E) | T*F | E+T

* **Exercise 7.1.1:** Find a grammar equivalent to

$$\begin{array}{lcl} S & \rightarrow & AB \mid CA \\ A & \rightarrow & a \\ B & \rightarrow & BC \mid AB \\ C & \rightarrow & aB \mid b \end{array}$$

with no useless symbols.

Variables: S, A, B and C

B is a useless symbol

S→CA
A→a
C→b

* **Exercise 7.1.2:** Begin with the grammar:

$$\begin{array}{lcl} S & \rightarrow & ASB \mid \epsilon \\ A & \rightarrow & aAS \mid a \\ B & \rightarrow & SbS \mid A \mid bb \end{array}$$

a) Eliminate ϵ-productions.

b) Eliminate any unit productions in the resulting grammar.

c) Eliminate any useless symbols in the resulting grammar.

d) Put the resulting grammar into Chomsky Normal Form.

Solution:
a) The given grammar has 1 €-Production: S→€
    S→ ASB | AB
    A→ aAS | a | aA
    B→ SbS | A | bb | bS | Sb | b

b) Unit Production in the resultant grammar: 1 (A→B)
    S→ ASB | AB
    A→ aAS | a | aA

B→SbS | aAS | a | aA | bb | bS | Sb | b

c) The resultant grammar has no useless symbols

S→ ASB | AB

A→ aAS | a | aA

B→SbS | aAS | a | aA | bb | bS | Sb | b

Chomsky Normal Form (CNF): A→BC and A→a

Consider: X→a and Y→b

S→ ASB | AB

A→ XAS | a | XA

B→SYS | XAS | a | XA | YY | YS | SY | b

Now S→ AC | AB

C→SB

A→XD | a | XA

D→AS

B→SE | XD | a | XA | YY | YS | SY| b

E→YS

CNF Grammar is

S→ AC | AB

C→SB

A→XD | a | XA

D→AS

B→SE | XD | a | XA | YY | YS | SY| b

E→YS

X→a

Y→b

Example 2:

Consider the grammar

$$\begin{array}{lll} I & \rightarrow & a \mid b \mid Ia \mid Ib \mid I0 \mid I1 \\ F & \rightarrow & I \mid (E) \\ T & \rightarrow & F \mid T * F \\ E & \rightarrow & T \mid E + T \end{array}$$

Convert the above grammar in CNF.

The given grammar has no €-productions
The grammar has three unit productions F→I, T→F, and E→T

I→ a | b | Ia | Ib | I0 |I1

F→ a | b | Ia | Ib | I0 |I1 | (E)

T→ a | b | Ia | Ib | I0 |I1 | (E) | T*F

E→ a | b | Ia | Ib | I0 |I1 | (E) | T*F | E+T

The grammar has no useless symbols
the new variables, we introduce:

$$\begin{array}{llll} A \to a & B \to b & Z \to 0 & O \to 1 \\ P \to + & M \to * & L \to (& R \to) \end{array}$$

Now
I→ a | b | IA | IB | IZ |IO

F→ a | b | IA | IB | IZ |IO | LER

T→ a | b | IA | IB | IZ |IO | LER | TMF

E→ a | b | IA | IB | IZ |IO | LER | TMF | EPT

The CNF Grammar is

I→ a | b | IA | IB | IZ |IO

F→ a | b | IA | IB | IZ |IO | LC1
C1→ER

T→ a | b | IA | IB | IZ |IO | LC1 | TC2
C2→ MF

E→ a | b | IA | IB | IZ |IO | LC1 | TC2 | EC3
C3→ PT

A→a	B→b	Z→0	O→1
P→+	M→*	L→(	R→)

CNF Form:: A→BC and A→a

GNF Form:: $A \rightarrow bD_1D_2D_3...D_n$ and A→b

Example: Convert the following CFG into GNF

S→ XY | Xn | p
X → mX | m
Y → Xn | o

Do proper substitutions

S→ mXY | mY | mXn | mn | p
X → mX | m
Y→ mXn | mn | o

Consider N→n

S→ mXY | mY | mXN | mN | p
X → mX | m
Y→ mXN | mN | o

GNF grammar is

S→ mXY | mY | mXN | mN | p
X → mX | m
Y→ mXN | mN | o
N→ n

Example 2: Consider the CFG

S→ Aa

A→ aA | bA | aAS convert the given CFG into GNF.

After Substitution

S→ aAa | bAa | aASa

A→ aA | bA | aAS

Assume X→a

GNF Grammar is:

S→ aAX | bAX | aASX

A→ aA | bA | aAS

X→a

Example 3: Convert the given CFG grammar into GNF

S→ABA

A→aA | €

B→bB | €

Given CFG has €-Productions, So eliminate them

S → ABA | BA | AB | AA | A | B

A→ aA | a

B→ bB | b

The resultant grammar has unit productions, so eliminate them

S → ABA | BA | AB | AA | aA | a | bB | b

A→ aA | a

B→ bB | b

The resultant grammar has no useless symbols

S → ABA | BA | AB | AA | aA | a | bB | b

A→ aA | a

B→ bB | b

Let us replace A by aA or a in S and B by bB or b in S

S→ aABA | bBA | aAB | aAA | aA | a | bB | b
A→ aA | a
B→ bB | b → GNF Grammar
Pumping Lemma for CFL:

Used for proving the given language is not Context Free Language.

Examples:

1. Find out whether the language **$L = \{x^n\ y^n\ z^n \mid n \geq 1\}$** is context free language or not.

Language L={ xyz, xxyyzz, xxxyyyzzz,...........}

Z is any string in L

Z = x x yy z z = UVWXY break this Z in to three parts

i) V= x and X= z
ii) |VWX| <=n |xyyz| <=n

iii) for i>=0

$U\ V^i\ W\ X^i\ Y = x\ x^i\ yy\ z^i\ z$

consider i=2 x xx yy zz z Not belongs to L

Hence, the given language is not CFL.

2. Prove the language **$L = \{a^i\ b^j\ c^k \mid i<j<k\}$ is not CFL.**

L= { abbccc, aabbbcccc, aaabbbbccccc,..................}

z= abbccc = uvwxy

i) v=b and x = c

ii) |vwx| = |bbcc| <=n

iii) for i>=0 , u v^i w x^i y

Consider i=0 a b^0 bc c^0 c

a bc c not belongs to L

Hence, the given language is not CFL.

PDA to CFG Conversion

Input: PDA transition functions

Output: Equivalent Grammar productions

Example:

PDA $P_N = (\{q\}, \{i, e\}, \{Z\}, \delta_N, q, Z)$

1. Start symbol of Grammar G: S→ [qzq]

2. for production $\delta_N(q, i, Z)$ contains (q, ZZ),

[qzq] → i [qzq][qzq]

3. From the fact that $\delta_N(q, e, Z)$ contains (q, ϵ), we have production

[qzq]→e

Example 2:

Convert the PDA P = ({q}, {0,1}, {X,Z,U} δ, q, Z) to a grammar where δ is given by

1. δ(q, €, z) = { (q, €)}
2. δ(q, 0, z) = { (q, zz)}
3. δ(q, 0, x) = { (q, xx)}
4. δ(q, 1, z) = { (q, uz)}
5. δ(q, 1, u) = { (q, uu)}
6. δ(q, 0, u) = { (q, €)}

7. δ(q, 1, x) = { (q, €)}

Solution: Start symbol S: S→ [qzq]   S→A
variables are [qzq]= A,   [qxq]=B,   [quq]=C
Grammar productions are:

1. δ(q, €, z) = { (q, €)}

    [qzq]→€   A→€

2. δ(q, 0, z) = { (q, zz)}

    [qzq]→ 0 [qzq][qzq]   A→ 0AA

3. δ(q, 0, x) = { (q, xx)}
    [qxq]→ 0 [qxq][qxq]   B→ 0BB

4. δ(q, 1, z) = { (q, uz)}
    [qzq]→ 1 [quq][qzq]   A→1CA

5. δ(q, 1, u) = { (q, uu)}
    [quq]→ 1 [quq][quq]   C→1CC

6. δ(q, 0, u) = { (q, €)}
    [quq]→ 0   C→0

7. δ(q, 1, x) = { (q, €)}
    [qxq]→1   B→1

* **Exercise 6.3.3:** Convert the PDA $P = (\{p, q\}, \{0, 1\}, \{X, Z_0\}, \delta, q, Z_0)$ to a CFG, if δ is given by:

1. $\delta(q, 1, Z_0) = \{(q, XZ_0)\}$.
2. $\delta(q, 1, X) = \{(q, XX)\}$.
3. $\delta(q, 0, X) = \{(p, X)\}$.
4. $\delta(q, \epsilon, X) = \{(q, \epsilon)\}$.
5. $\delta(p, 1, X) = \{(p, \epsilon)\}$.
6. $\delta(p, 0, Z_0) = \{(q, Z_0)\}$.

Start symbol S: S→ [qzp] / [qzq]

From transition rule 1:

[qzq] → 1 [qxq][qzq]
[qzq] → 1 [qxp][pzq]
[qzp] → 1 [qxq][qzp]
[qzp] → 1 [qxp][pzp]

From transition rule 2:

[qxq] → 1 [qxq][qxq]
[qxq] → 1 [qxp][pxq]
[qxp] → 1 [qxq] [qxp]
[qxp] → 1 [qxp] [pxp]

From transition rule 3:

[qxq] → 0 [pxq]

[qxp] → 0 [pxp]

From transition rule 4:

[qxq] → €

From transition rule 5:

[pxp] → 1

From transition rule 6:

[pzq] → 0 [qzq]
[pzp] → 0 [qzp]