UNIT 4 QUESTION ANSWERS

1. Remove the left recursion from the following grammar

$$E \rightarrow E + T \mid T$$
$$T \rightarrow T{*}F \mid F$$
$$F \rightarrow id \mid (E)$$

Solution: In the grammar there are two immediate left recursions $E \rightarrow E + T$ and $T \rightarrow T * F$. By using Moore's proposal, the left recursion $E \rightarrow E + T$ is removed as

$$E \rightarrow TE' \text{ and } E' \rightarrow +TE'/\in$$

The left recursion $T \rightarrow T{*}F$ is removed as

$$T \rightarrow FT'$$
$$T' \rightarrow {*}FT' \mid \in$$

The CFG after removing the left recursion becomes

$$E \rightarrow TE'$$
$$E' \rightarrow +TE' \mid \in$$
$$T \rightarrow FT'$$
$$T' \rightarrow {*}FT' \mid \in$$
$$F \rightarrow id \mid (E)$$

2. Remove the left recursion from the following grammar.

$$S \rightarrow Aa \mid b$$
$$A \rightarrow Sc \mid d$$

***Solution*:** The grammar has indirect left recursion. Rename S as A_1 and A as A_2. The grammar is

$$A1 \rightarrow A2a/b$$
$$A2 \rightarrow A1c/d.$$

For i = 1, j = 1, there is no production in the form $A_1 \rightarrow A_1\alpha$.

For i = 2, j = 1, there is a production in the form $A_2 \rightarrow A_1\alpha$. The production is $A_2 \rightarrow A_1c$. According to the algorithm for removal of indirect left recursion, the production becomes

$$A_2 \rightarrow A_2ac/bc$$

This left recursion for the production $A_2 \rightarrow A_2ac/bc/d$ is removed and the production rules are

$A_2 \rightarrow bcA'_2/dA'_2$

$A'_2 \rightarrow acA'_2/\in$

The actual non-left recursive grammar is

$S \rightarrow Aa/b$
$A \rightarrow bcA'/dA'$
$A' \rightarrow acA'/\in$.

3. Remove the useless symbols from the given CFG.

$S \rightarrow AC \quad S \rightarrow BA \quad C \rightarrow CB \quad C \rightarrow AC$
$A \rightarrow a$
$B \rightarrow aC/b$

Solution: There are two types of useless symbols: non-generating and non-reachable symbols. First, we are finding non-generating symbols.

Those symbols which do not produce any terminal string are non-generating symbols.

Here, C is a non-generating symbol as it does not produce any terminal string.

So, we have to remove the symbol C. To remove C, all the productions containing C as a symbol (LHS or RHS) must be removed. By removing the productions, the minimized grammar will be

$S \rightarrow BA$
$A \rightarrow a$
$B \rightarrow b$

Now, we have to fi nd non-reachable symbols, the symbols which cannot be reached at any time starting from the start symbol. There is no non-reachable symbol in the grammar. So, the minimized form of the grammar by removing useless symbols is

$S \rightarrow BA$
$A \rightarrow a$

$B \rightarrow b$

4. Remove the unit production from the following grammar.

$$S \rightarrow AB,\ A \rightarrow E,\ B \rightarrow C,\ C \rightarrow D,\ D \rightarrow b,\ E \rightarrow a$$

Solution: In this grammar unit, the productions are $A \rightarrow E$, $B \rightarrow C$, and $C \rightarrow D$. If we want to remove the unit production $A \rightarrow E$ from the grammar, first we have to find a non-unit production in the form $E \rightarrow$ {Some string of terminal or non-terminal or both}. There is a production $E \rightarrow a$. So, the production $A \rightarrow a$ will be added to the production rule. And the modified production rule will be

$$S \rightarrow AB,\ A \rightarrow a,\ B \rightarrow C,\ C \rightarrow D,\ D \rightarrow b,\ E \rightarrow a$$

For removing another unit production $B \rightarrow C$, we have to find a non-unit production

$C \rightarrow$ {Some string of terminal or non-terminal or both}. But there is no such production found.

For removing a unit production $C \rightarrow D$ we have found a non-unit production $D \rightarrow b$. So, the production $C \rightarrow b$ will be added to the production rule set and $C \rightarrow D$ will be removed. The modified production rule will be

$$S \rightarrow AB,\ A \rightarrow a,\ B \rightarrow C,\ C \rightarrow b,\ D \rightarrow b,\ E \rightarrow a$$

Now the unit production $B \rightarrow C$ will be removed by $B \rightarrow b$ as there is a non-unit production $C \rightarrow b$.

The modified production rule will be

$$S \rightarrow AB,\ A \rightarrow a,\ B \rightarrow b,\ C \rightarrow b,\ D \rightarrow b,\ E \rightarrow a$$

[This is the grammar without unit production, but it is not minimized. There are useless symbols (nonreachable symbols) C, D, and E. So, if we want to minimize the grammar, these symbols will be removed. By removing the useless symbols, the grammar will be $S \rightarrow AB$, $A \rightarrow a$, $B \rightarrow b$.]

5. Remove the unit production from the following grammar.

$$S \rightarrow aX/Yb/Y$$
$$X \rightarrow S$$
$$Y \rightarrow Yb/b$$

***Solution*:** The unit productions in the grammar are S → Y and X → S. To remove unit production S → Y, we have to find a non-unit production where Y → {Some string of terminal or non-terminal or both}. There is a non-unit production Y → Yb/b. The unit production S → Y will be removed by including production S → Yb/b in the production.

The modified production rules will be

S → aX/Yb/b
X → S
Y → Yb/b

In the previous production rules, there is a unit production X → S. This production can be removed by including the production X → aX/Yb/b to the production rules. The modified production rules will be

S → aX/Yb/b
X → aX/Yb/b
Y → Yb/b

(This is also minimized grammar.)

6. Remove the unit production from the following grammar.

S → AA
A → B/BB
Y → abB/b/bb

***Solution*:** In the previous grammar, there is a unit production A → B. To remove the unit production,

we have to find a non-unit production Y → {Some string of terminal or non-terminal or both}.

There is a non-unit production B → abB/b/bb . So, the unit production A → B can be removed by including the production A → abB/b/bb to the production rule. The modified production rule will be

S → AA
A → BB/abB/b/bb
B → abB/b/bb.

(This is also minimized grammar.)

7. Remove the ∈ production from the following grammar.

$$S \rightarrow aA$$
$$A \rightarrow b/\in$$

Solution: In the previous production rules, there is a null production $A \rightarrow \in$. The grammar does not produce null string as a language, and so this null production can be removed.

According to fi rst step we have to look for all productions whose right side contains A. There is one $S \rightarrow aA$.

According to the second step, we have to replace each occurrence of 'A' in each of the productions.

There is only one occurrence of 'A' in $S \rightarrow aA$. So, after replacing, it will become $S \rightarrow a$. According to step three, these productions will be added to the grammar.

So, the grammar will be

$$S \rightarrow aA/a$$
$$A \rightarrow b$$

Now, in the new production rules, there is no null production.

8. Remove the ∈ production from the following grammar.

$$S \rightarrow aX/bX$$
$$X \rightarrow a/b/\in$$

***Solution*:** In the previous grammar, there is a null production $X \rightarrow \in$. In the language set produced by the grammar, there is no null string, and so this null production can be removed.

For removing this null production, we have to look for all the productions whose right side contains X. There are two such productions in the grammar $S \to aX$ and $S \to bX$.

We have to replace each occurrence of 'X' in each of the productions to obtain a non-null production. In both the productions, X has occurred only once. So, after replacing, the productions will become $S \to a$ and $S \to b$, respectively.

These productions must be added to the grammar to keep the language generating power the same. So, after adding these productions, the grammar will be

$$S \to aX/bX/a/b$$
$$X \to a/b$$

9. Convert the following grammar into CNF.

$S \to bA/aB$
$A \to bAA/aS/a$

$B \to aBB/bS/a$

Solution: The productions $S \to bA$, $S \to aB$, $A \to bAA$, $A \to aS$, $B \to aBB$, $B \to bS$ are not in CNF.

So, we have to convert these into CNF. Let us replace terminal 'a' by a non-terminal C_a and terminal 'b' by a non-terminal C_b. Hence, two new productions will be added to the grammar

$$C_a \to a \text{ and } C_b \to b$$

By replacing a and b by new non-terminals and including the two productions, the modifi ed grammar will be

$S \to C_bA/C_aB$

$A \to C_bAA/C_aS/a$

$B \to C_aBB/C_bS/a$

$C_a \to a$

$C_b \to b$

In the modified grammar, all the productions are not in CNF. The productions $A \to C_bAA$ and $B \to C_aBB$ are not in CNF, because they contain more than two non-terminals at the RHS. Let us replace AA by a new non-terminal D and BB by another new non-terminal E. Hence, two new productions will be added to the grammar D → AA and E → BB. So, the new modified grammar will be

$S \rightarrow C_bA/C_aB$

$A \rightarrow C_bD/C_aS/a$

$B \rightarrow C_aE/C_aS/a$

$D \rightarrow AA$

$E \rightarrow BB$

$C_a \rightarrow a$

$C_b \rightarrow b$

10. Convert the following grammar into CNF.

$$E \rightarrow E + E$$
$$E \rightarrow E * E$$
$$E \rightarrow id \text{ where } \Sigma = \{ +, *, id\}.$$

Solution: Except E → id, all the other productions of the grammar are not in CNF. In the grammar E → E + E and E → E * E, there are two terminals + and *. Take two non-terminals C and D for replacing + and *, respectively. Two new productions will be added to the grammar. By replacing + and *, the modified production rules will be

E → ECE
E → EDE
E → id

C → +

D → *

In the production rules E → ECE and E → EDE, there are three non-terminals. Replace EC by another non-terminal F and ED by G. So, two new productions F → EC and G → ED will be added to the grammar. By replacing and adding the new productions, the modified grammar will be

E → FE
E → GE
E → id
C → +
D → *
F → EC

$G \rightarrow ED$

In the previous grammar, all the productions are in the form of non-terminal → string of exactly two non-terminals or non-terminal → single terminal. So, the grammar is in CNF.

11. Convert the following grammar into GNF.

$S \rightarrow AA/a \quad A \rightarrow SS/b$

Solution:

Step I: There are no unit productions and no null production in the grammar. The given grammar is in CNF.

Step II: In the grammar, there are two non-terminals S and A. Rename the non-terminals as A_1 and A_2, respectively. The modifi ed grammar will be

$$A_1 \rightarrow A_2A_2/a$$
$$A_2 \rightarrow A_1A_1/b$$

Step III: In the grammar, $A_2 \rightarrow A_1A_1$ is not in the format $A_i \rightarrow A_jV$ where $i \leq j$. Replace the leftmost A_1 at the RHS of the production $A_2 \rightarrow A_1A_1$. After replacing the modified A_2, production will be

$$A_2 \rightarrow A_2A_2A_1/aA_1/b$$

The production $A_2 \rightarrow aA_1/b$ is in the format $A \rightarrow \beta_i$ and the production $A_2 \rightarrow A_2A_2A_1$ is in the format of $A \rightarrow A\alpha_j$ So, we can introduce a new non-terminal B_2 and the modified A_2 production will be (accord-
ing to Lemma II)

$$A2 \rightarrow aA1/b$$
$$A_2 \rightarrow aA_1B_2$$
$$A_2 \rightarrow bB_2$$

And the B_2 productions will be

$$B_2 \rightarrow A_2A_1$$
$$B_2 \rightarrow A_2A_1B_2$$

Step IV: All A_2 productions are in the format of GNF. In the production $A_1 \rightarrow A_2A_2/a$, $A \rightarrow a$ is in the prescribed format. But the production $A_1 \rightarrow A_2A_2$ is not in the format of GNF. Replace the leftmost A_2 at the RHS of the production by the previous A_2 productions. The modified A_1 productions will be

$$A_1 \rightarrow aA_1A_2/bA_2/aA_1B_2A_2/bB_2A_2$$

The B_2 productions are not in GNF. Replace the leftmost A_2 at the RHS of the two productions by the A_2 productions. The modified B_2 productions will be

$$B_2 \rightarrow aA_1A_1/bA_1/aA_1B_2A_1/bB_2A_1$$
$$B2 \rightarrow aA1A1B2/bA1B2/aA1B2A1B2/bB2A1B2$$

For the given CFG, the GNF will be

$$A1 \rightarrow aA_1A_2/bA_2/aA_1B_2A_2/bB_2A_2/a$$
$$A2 \rightarrow aA_1/b/aA_1B_2/bB_2$$
$$B2 \rightarrow aA_1A_1/bA_1/aA_1B_2A_1/bB_2A_1$$
$$B2 \rightarrow aA_1A_1B_2/bA_1B_2/aA_1B_2A_1B_2/bB_2A_1B_2$$

12. Convert the following CFG into GNF.

$S \rightarrow XY$
$X \rightarrow YS/b$
$Y \rightarrow SX/a$

Solution:

Step I: In the grammar, there is no null production and no unit production. The grammar also is in CNF.

Step II: In the grammar, there are three non-terminals S, X, and Y. Rename the non-terminals as A_1, A_2, and A_3, respectively. After renaming, the modified grammar will be

$$A_1 \rightarrow A_2A_3$$

$A_2 \rightarrow A_3A_1/b$

$A_3 \rightarrow A_1A_2/a$

Step III: In the grammar, the production $A_3 \rightarrow A_1A_2$ is not in the format $A_i \rightarrow A_jV$ where $i \leq j$.

Replace the leftmost A_1 at the RHS of the production $A_3 \rightarrow A_1A_2$ by the production $A_1 \rightarrow A_2A_3$. The production will become $A_3 \rightarrow A_2A_3A_2$, which is again not in the format of $A_i \rightarrow A_jV$ where $i \leq j$. Replace the leftmost A_2 at the RHS of the production $A_3 \rightarrow A_2A_3A_2$ by the production $A_2 \rightarrow A_3A_1/b$. The modified A_3 production will be

$$A_3 \rightarrow A_3A_1A_3A_2/bA_3A_2/a$$

The production $A_3 \rightarrow bA_3A_2/a$ is in the format of $A \rightarrow \beta_i$ and the production $A_3 \rightarrow A_3A_1A_3A_2$ is in the format of $A \rightarrow A\alpha_j$ So, we can introduce a new non-terminal B and the modified A_3 production will be (according to Lemma II)

$A_3 \rightarrow bA_3A_2$

$A_3 \rightarrow a$

$A_3 \rightarrow bA_3A_2B$

$A_3 \rightarrow aB$

And B productions will be

$B \rightarrow A_1A_3A_2$

$B \rightarrow A_1A_3A_2B$

Step IV: All the A_3 productions are in the specified format of GNF.

The A_2 production is not in the specified format of GNF. Replacing A_3 productions in A_2 productions, the

modified A_2 production becomes

$A_2 \rightarrow bA_3A_2A_1/aA_1/bA_3A_2BA_1/aBA_1/b$

Now, all the A_2 productions are in the prescribed format of GNF.

The A_1 production is not in the prescribed format of GNF. Replacing A_2 productions in A_1, the modified A_1 productions will be

$A_1 \rightarrow bA_3A_2A_1A_3/aA_1A_3/bA_3A_2BA_1A_3/aBA_1A_3/bA_3$

All the A_1 productions are in the prescribed format of GNF.

But the B productions are still not in the prescribed format of GNF. By replacing the leftmost A_1 at the RHS of the B productions by A_1 productions, the modified B productions will be

B → bA3A2A1A3A3A2/aA1A3A3A2/bA3A2BA1A3A3A2/aBA1A3A3A2/bA3A3A2

B → bA3A2A1A3A3A2B/aA1A3A3A2B/bA3A2BA1A3A3A2B/aBA1A3A3A2B/bA3A3A2B.

Now, all the B productions of the grammar are in the prescribed format of GNF. So, for the given CFG, the GNF will be

$A_1 \to bA_3A_2A_1A_3/aA_1A_3/bA_3A_2BA_1A_3/aBA_1A_3/bA_3$

$A_2 \to bA_3A_2A_1/aA_1/bA_3A_2BA_1/aBA_1/b$

B → bA3A2A1A3A3A2/aA1A3A3A2/bA3A2BA1A3A3A2/aBA1A3A3A2/bA3A3A2

B → bA3A2A1A3A3A2B/aA1A3A3A2B/bA3A2BA1A3A3A2B/aBA1A3A3A2B/bA3A3A2B

13. Show that L = $\{a^nb^nc^n$ where $n \geq 1\}$ is not context free.

Solution:

Step I: Assume that the language set L is a CFL. Let n be a natural number obtained by using the pumping lemma.

Step II: Let $z = a^nb^nc^n$. So, $|z| = 3n > n$. According to the pumping lemma for CFL, we can write z = uvwxy, where $|vx| \geq 1$.

Step III: uvwxy = $a^n b^n c^n$. As $1 \leq | vx | \leq n$, ($| vwx | \leq n$, so $| vx | \leq n$) v or x cannot contain all the three symbols a, b, and c. So, v or x will be in any of the following forms.

1. Contain only a and b, i.e., in the form $a^i b^j$.
2. Or contain only b and c, i.e., in the form $b^i c^j$.
3. Or contain only the repetition of any of the symbols among a, b, and c.

Let us take the value of k as 2. v^2 or x^2 will be in the form $a^i b^i a^i b^i$ (as v is a string here aba is not equal to a^2b or ba^2) or $b^i c^j$ $b^i c^j$. So, uv^2wx^2y cannot be in the form $a^m b^m c^m$. So, $uv^2wx^2y \notin L$.

If v or x contains repetition of any of the symbols among a, b, and c, then v or x will be any of the form of a^i, b^i, or c^i. Let us take the value of k as 0. uv^0wx^0y = uwy. In the string, the number of occurrences of one of the other two symbols in uvy is less than n. So, $uv^2wx^2y \notin L$.

14. Prove that the language $L = \{a^{i2}/i \geq 1 \}$ is not context free.

Solution:

Step I: Assume that the language set L is a CFL. Let n be a natural number obtained by using the pumping lemma.

Step II: Let $z = a^{i2}$. So, $| z | = 2^i$. Let $2^i > n$. According to the pumping lemma for CFL, we can write z = uvwxy, where $| vx | \geq 1$ and $| vwx | \leq n$.

Step III: The string z contains only ‘a’, and so v and x will also be a string of only ‘a’. Let $v = a^p$ and $x = a^q$, where $(p + q) \geq 1$. Since $n \geq 0$ and $uvwxy = 2^i$, $| uv^nwx^ny | = | uvwxy | + | v^{n-1}x^{n-1} | = 2^i + (p + q)(n - 1)$. As $uv^nwx^ny \in L$, $| uv^nwx^ny |$ is also a power of 2, say 2^j.

$(p + q)(n - 1) = 2^j - 2^i \Rightarrow (p + q)(n - 1) + 2^i = 2^j$

$\Rightarrow (p + q)2^{i+1} + 2^i = 2^j$

$$\Rightarrow 2^i (2(p + q) + 1) = 2^j$$

(p + q) may be even or odd, but 2(p + q) is always even. However, 2(p + q) + 1 is odd, which cannot be a power of 2. Thus, L is not context free

15. Prove that $L = \{0^p/\text{where p is prime}\}$ is not a CFL.

Solution:

Step I: Suppose L = L(G) is context free. Let n be a natural number obtained from the pumping lemma for CFL.

Step II: Let p be a number > n, $z = 0^p \in L$. By using the pumping lemma for CFL,

we can write z = uvwxy, where $| vx | \geq 1$ and $| vwx | \leq n$.

Step III: Let k = 0. From the pumping lemma for CFL, we can write uv^0wx^0y, i.e., $uwy \in L$. As uvy is in the form 0^p, where p is prime, | uwy | is also a prime number. Let us take it as q. Let | vx | = r. Then, $| uv^qwx^qy | = q + qr = q(1 + r)$. This is not prime as it has factors q(1 + r) including 1 and q(1 + r). So, $uv^qwx^qy \notin L$. This is a contradiction. Therefore, L is not context free.

16. Design a TM to accept the language $L = \{a^nb^n, n \geq 1\}$. Show an ID for the string 'aaabbb' with tape symbols.

Solution: The string consists of two types of input alphabets, 'a' and 'b'. The number of 'a' is equal to the number of 'b'. All the 'a's will come before 'b'. In the language set, there is at least one 'a' and one 'b'. The TM can be designed as follows.

When the leftmost 'a' is traversed, that 'a' is replaced by X and the head moves to one right. The transitional function is

$$\delta(q_0, a) \rightarrow (q_1, X, R)$$

Then, the machine needs to search for the leftmost 'b'. Before that 'b', there exist (n − 1) numbers of 'a'. Those 'a' are traversed by

$$\delta(q_1, a) \rightarrow (q_1, a, R)$$

When the leftmost 'b' is traversed, the state is q_1. That 'b' is replaced by Y, the state is changed to q_2, and the head is moved to one left. The transitional functional is

$$\delta(q_1, b) \rightarrow (q_2, Y, L)$$

Then, it needs to search for the second 'a' starting from the left. The fi rst 'a' is replaced by X, which means the second 'a' exists after X. So, it needs to search for the rightmost 'X'. After traversing the leftmost 'b', the head moves to the left to fi nd the rightmost X. Before that, it has to traverse 'a'. The transitional function is

$$\delta(q_2, a) \rightarrow (q_2, a, L)$$

After traversing all the 'a' we get the rightmost 'X'. Traversing the X the machine changes its state from q_2 to q_0 and the head moves to one right. The transitional function is

$$\delta(q_2, X) \rightarrow (q_0, X, R)$$

From the traversal of the second 'b' onwards, the machine has to traverse 'Y'. The transitional function is

$$\delta(q_1, Y) \rightarrow (q_1, Y, R)$$

Similarly, after traversing 'b', the machine has to traverse some Y to get the rightmost 'X'.

$$\delta(q_2, Y) \rightarrow (q_2, Y, L)$$

When all the 'a's are traversed, the state is q_0, because before that state was q_2 and the input was X. The head moves to one right and gets a Y. Getting a Y means that all the 'a's are traversed and the same number of 'b's are traversed. Traversing right, if at last the machine gets no 'b' but a blank 'B', then the machine halts. The transitional functions are

$\delta(q_0, Y) \rightarrow (q_3, Y, R)$ $\delta(q_3, Y) \rightarrow (q_3, Y, R)$

$\delta(q_3, B) \rightarrow (q_4, B, H)$

ID for the String 'aaabbb'

B	a	a	a	b	b	b	B

First, the tape symbols are

$\delta(q_0, a) \rightarrow (q_1, Xaabbb, R)$

B	X	a	a	b	b	b	B

$\delta(q_1, a) \rightarrow (q_1, Xaabbb, R)$

B	X	a	a	b	b	b	B

$\delta(q_1, a) \rightarrow (q_1, Xaabbb, R)$

B	X	a	a	b	b	b	B

$\delta(q_1, b) \rightarrow (q_2, XaaYbb, L)$

B	X	a	a	Y	b	b	B

$\delta(q_2, a) \rightarrow (q_2, XaaYbb, L)$	B	X	a	a	Y	b	b	B
$\delta(q_2, a) \rightarrow (q_2, XaaYbb, L)$	B	X	a	a	Y	b	b	B
$\delta(q_2, X) \rightarrow (q_0, XaaYbb, R)$	B	X	a	a	Y	b	b	B
$\delta(q_0, a) \rightarrow (q_1, XXaYbb, R)$	B	X	X	a	Y	b	b	B
$\delta(q_1, a) \rightarrow (q_1, XXaYbb, R)$	B	X	X	a	Y	b	b	B
$\delta(q_1, Y) \rightarrow (q_1, XXaYbb, R)$	B	X	X	a	Y	b	b	B
$\delta(q_1, b) \rightarrow (q_2, XXaYYb, L)$	B	X	X	a	Y	Y	b	B
$\delta(q_2, Y) \rightarrow (q_2, XXaYYb, L)$	B	X	X	a	Y	Y	b	B
$\delta(q_2, a) \rightarrow (q_2, XXaYYb, L)$	B	X	X	a	Y	Y	b	B
$\delta(q_2, X) \rightarrow (q_0, XXaYYb, R)$	B	X	X	a	Y	Y	b	B
$\delta(q_0, a) \rightarrow (q_1, XXXYYb, R)$	B	X	X	X	Y	Y	b	B
$\delta(q_1, Y) \rightarrow (q_1, XXXYYb, R)$	B	X	X	X	Y	Y	b	B

$\delta(q_1, Y) \rightarrow (q_1,$ XXXYYb, R)	B	X	X	X	Y	Y	b	B	
$\delta(q_1, b) \rightarrow (q_2,$ XXXYYY, L)	B	X	X	X	Y	Y	Y	B	
$\delta(q_2, Y) \rightarrow (q_2,$ XXXYYY,	B	X	X	X	Y	Y	Y	B	L)
$\delta(q_2, Y) \rightarrow (q_0,$ XXXYYY,	B	X	X	X	Y	Y	Y	B	R)
$\delta(q_0, Y) \rightarrow (q_3,$ XXXYYY,	B	X	X	X	Y	Y	Y	B	R)
$\delta(q_3, Y) \rightarrow (q_3,$ XXXYYY,	B	X	X	X	Y	Y	Y	B	R)
$\delta(q_3, Y) \rightarrow (q_3,$ XXXYYY,	B	X	X	X	Y	Y	Y	B	R)
$\delta(q_3, Y) \rightarrow (q_3,$ XXXYYY,	B	X	X	X	Y	Y	Y	B	R)
$\delta(q_3, B) \rightarrow (q_3,$ XXXYYY,	B	X	X	X	Y	Y	Y	B	H)

17. Design a TM to accept the language

$$L = \{WW^R, \text{ where } W \in (a, b)^+\}$$

Show an ID for the string ‘abaaaaba’ with tape symbols.

Solution: W^R is the reverse of W. If W starts with ‘a’ or ‘b’, W^R ends with ‘a’ or ‘b’, respectively. If W ends with ‘a’ or ‘b’, W^R starts with ‘a’ or ‘b’, respectively. The TM can be designed as follows.

If the string W starts with ‘a’, upon traversal, that ‘a’ is replaced by X with a state change from q_0 to q_1, and the head moves to one right. The transitional function is

$$\delta(q_0, a) \rightarrow (q_1, X, R)$$

W^R ends with ‘a’ if W starts with ‘a’. The machine needs to search the end ‘a’ of W^R. Before that, the machine needs to traverse the end symbols of W and the beginning symbols of W^R. The transitional functions are

$$\delta(q_1, a) \rightarrow (q_1, a, R) \qquad \delta(q_1, b) \rightarrow (q_1, b, R)$$

After the end symbol of W^R, there exists the blank symbol B. In the traversal process, if the machine gets a B, it traverses back to the left side and gets the end symbol of W^R. The transitional functions are

$$\delta(q_1, B) \rightarrow (q_2, B, L) \qquad \delta(q_2, a) \rightarrow (q_3, X, L)$$

Now, the machine needs to search for the second symbol of W. Before that, it has to traverse the beginning symbols of W^R and the end symbols of W. The transitional functions are

$$\delta(q_3, a) \rightarrow (q_3, a, L) \qquad \delta(q_3, b) \rightarrow (q_3, b, L)$$

When the machine gets the rightmost X, it recognizes that the next symbol of W exists after that ‘X’. The transitional function is

$$\delta(q_3, X) \rightarrow (q_0, X, R)$$

If the string W starts with ‘b’, the transitions are the same as the previous one but with some states changed. The transitional functions are

$$\delta(q_0, a) \rightarrow (q_4, Y, R) \qquad \delta(q_4, a) \rightarrow (q_4, a, R) \qquad \delta(q_4, b) \rightarrow (q_4, b, R)$$

$$\delta(q_4, B) \rightarrow (q_5, B, L) \qquad \delta(q_5, b) \rightarrow (q_6, Y, L) \qquad \delta(q_6, a) \rightarrow (q_6, a, L)$$

$$\delta(q_6, b) \rightarrow (q_6, b, L) \qquad \delta(q_6, Y) \rightarrow (q_0, Y, R)$$

After the fi rst traversal, i.e., from the second traversal onwards, the machine need not traverse up to the end of W^R. In state q_1 (W starts with ‘a’) or q_4 (W starts with ‘b’), if the machine gets X or Y, it traverses back to the left to point the rightmost ‘a’ or ‘b’. The transitional functions are

$$\delta(q_1, X) \rightarrow (q_2, X, L) \qquad \delta(q_1, Y) \rightarrow (q_2, Y, L)$$

$$\delta(q_4, X) \rightarrow (q_5, X, L) \qquad \delta(q_4, Y) \rightarrow (q_5, Y, L)$$

When all the symbols of W and W^R are traversed, the machine gets an X or Y in the state q_0. The machine halts if in state q_0 it gets an X or Y. The transitional functions are

$\delta(q_0, X) \rightarrow (q_f, X, H)$ $\delta(q_0, Y) \rightarrow (q_f, Y, H)$

The transitional functions can be given in a tabular format as follows.

State	a	B	B	X	Y
q0	(q_1, X, R)	(q_4, Y, R)	–	(q_f, X, H)	(q_f, Y, H)
q1	(q_1, a, R)	(q_1, b, R)	(q_2, B, L)	(q_2, X, L)	(q_2, Y, L)
q2	(q_3, X, L)	–	–	–	–
q3	(q_3, a, L)	(q_3, b, L)	–	(q_0, X, R)	–
q4	(q_4, a, R)	(q_4, b, R)	(q_5, B, L)	(q_5, X, L)	(q_5, Y, L)
q5		(q_6, Y, L)	–	–	–
q6	(q_6, a, L)	(q_6, b, L)	–	–	(q_0, Y, R)

ID for the String 'abaaaaba': (Here, the symbols in bold represent the read–write head position.)

(q_0, **a**baaaaba) → (q_1, X**b**aaaaba) → (q_1, Xb**a**aaaba) → (q_1, Xba**a**aaba) → (q_1, Xbaa**a**aba)

→ (q_1, Xbaaa**a**ba) → (q_1, Xbaaaa**b**a) → (q_1, Xbaaaab**a**) → (q_1, Xbaaaaba**B**) → (q_2, Xbaaaab**a**B)

→ (q_3, Xbaaaa**b**X) → (q_3, Xbaaaa**b**X) → (q_3, Xbaaa**a**bX) → (q_3, Xbaa**a**abX) → (q_3, Xba**a**aabX)

→ (q_3, Xb**a**aaabX) → (q_3, X**b**aaaabXB) → (q_3, **X**baaaabX) → (q_0, X**b**aaaabX) → (q_4, XY**a**aaabX)

→ (q_4, XYa**a**aabX) → (q_4, XYaa**a**abX) → (q_4, XYaaa**a**bX) → (q_4, XYaaaa**b**X) → (q_4, XYaaaab**X**)

→ (q_5, XYaaaa**b**X) → (q_6, XYaaa**a**YX) → (q_6, XYaa**a**aYX) → (q_6, XYa**a**aaYX) → (q_6, XY**a**aaaYX)

→ (q_0, XY**a**aaaYX) → (q_1, XYX**a**aaYX) → (q_1, XYXa**a**aYX) → (q_1, XYXaa**a**YX) → (q_1, XYXaaa**Y**X)

→ (q_2, XYXaa**a**YX) → (q_3, XYXa**a**XYX) → (q_3, XYX**a**aXYX) → (q_3, XY**X**aaXYX) → (q_0, XYX**a**aXYX)

→ (q_1, XYXX**a**XYX) → (q_1, XYXXa**X**YX) → (q_2, XYXX**a**XYX) → (q_3, XYX**X**XXYX)

→ (q_0, XYXX**X**XYX) → (q_f, XYXXXXYX) [Halt]

18. Design a TM to accept the language L = {set of all palindromes over a, b}. Show the IDs for the null string, 'a', 'aba', and 'baab'.

Solution: Palindromes are of two types:

i) Odd palindromes, where the number of characters is odd ii) Even palindrome, where the number of characters is even. A null string is also a palindrome.

A string which starts with 'a' or 'b' must end with 'a' or 'b', respectively, if the string is a palindrome.

If a string starts with 'a', that 'a' is replaced by a blank symbol 'B' upon traversal with a state change from q_1 to q_2 and the right shift of the read–write head. The transitional function is

$$\delta(q_1, a) \rightarrow (q_2, B, R)$$

Then, the machine needs to search for the end of the string. The string must end with 'a', and that 'a' exists before a blank symbol B at the right hand side. Before getting that blank symbol, the machine needs to traverse all the remaining 'a' and 'b' of the string. The transitional functions are

$$\delta(q_2, a) \rightarrow (q_2, a, R) \qquad \delta(q_2, b) \rightarrow (q_2, b, R) \qquad \delta(q_2, B) \rightarrow (q_3, B, L)$$

$$\delta(q_3, a) \rightarrow (q_4, B, L)$$

The machine now needs to search for the second symbol (from the starting) of the string. That symbol exists after the blank symbol B, which is replaced at the fi rst. Before that the machine needs to traverse the remaining 'a' and 'b' of the string. The transitional functions are

$$\delta(q_4, a) \rightarrow (q_4, a, L) \qquad \delta(q_4, b) \rightarrow (q_4, b, L) \qquad \delta(q_4, B) \rightarrow (q_1, B, R)$$

If the string starts with 'b', the transitional functions are the same as 'a' but some states are changed. The transitional functions are

$$\delta(q_1, b) \rightarrow (q_5, B, R) \qquad \delta(q_5, a) \rightarrow (q_5, a, R) \qquad \delta(q_5, b) \rightarrow (q_5, b, R)$$

$$\delta(q_5, B) \rightarrow (q_6, B, L) \qquad \delta(q_6, b) \rightarrow (q_4, B, L)$$

When all 'a' and 'b' are traversed and replaced by B, the states may be one of q_3 or q_6, if the last symbol traversed is 'a' or 'b', respectively. Transitional functions for acceptance are

$$\delta(q_3, B) \rightarrow (q_7, B, H) \qquad \delta(q_6, B) \rightarrow (q_7, B, H)$$

A null string is also a palindrome. On the tape, a null symbol means blank B. In state q_1, the machine gets the symbol. The transitional function is

$$\delta(q_1, B) \rightarrow (q_7, B, H)$$

The transitional functions in tabular form are represented as follows.

State	A	b	B
q1	(q_2, B, R)	(q_5, B, R)	(q_7, B, H)
q2	(q_2, a, R)	(q_2, b, R)	(q_3, B, L)
q3	(q_4, B, L)	–	(q_7, B, H)
q4	(q_4, a, L)	(q_4, b, L)	(q_1, B, R)
q5	(q_5, a, R)	(q_5, b, R)	(q_6, B, L)
q6	–	(q_4, B, L)	(q_7, B, H)

ID for Null String

$$(q_1, B) \rightarrow (q_7, B) \text{ [Halt]}$$

ID for the String 'a'

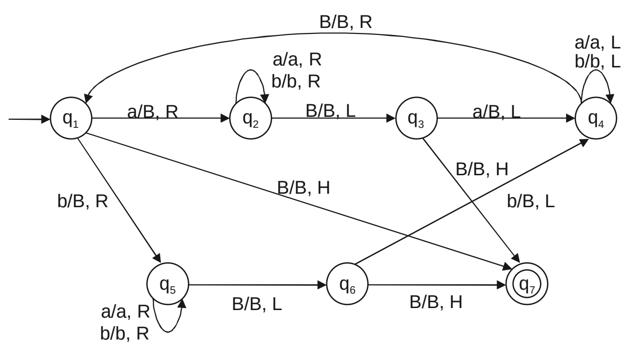
(q_1, **a**B) → (q_2, B**B**) → (q_3, **B**B) → (q_7, BB) [Halt]

ID for the String 'aba'

(q_1, **a**baB) → (q_2, B**b**aB) → (q_2, Bb**a**B) → (q_2, Bba**B**) → (q_3, Bb**a**B) → (q_4, B**b**BB) → (q_4, **B**bBB)
→ (q_1, BbBB) → (q_1, BBBB) → (q_7, BBBB) [Halt]

ID for the String 'baab'

(q_1, **b**aabB) → (q_5, B**a**abB) → (q_5, Ba**a**bB) → (q_5, Baa**b**B) → (q_5, Baab**B**) → (q_6, Baa**b**B) → (q_4, Ba**a**BB) → (q_4, B**a**aBB) → (q_4, **B**aaBB) → (q_1, B**a**aBB) → (q_2, BB**a**BB) → (q_2, BBa**B**B) → (q_3, BB**a**BB) → (q_3, B**B**BBB) → (q_7, BBBBB) [Halt].



19. Design a TM to accept the language L = $a^n b^n c^n$, where n ≥ 1.

Solution: This is a context-sensitive language. The language consists of a, b, and c. Here, the number of 'a' is equal to the number of 'b' which is equal to the number of 'c'. n number 'c' is followed by n number of 'b' which are followed by n number of 'a'. The TM is designed as follows. For each 'a', search for 'b' and 'c' and again traverse back to the left to search for the leftmost 'a'. 'a' is replaced by 'X', 'b' is replaced by 'Y', and 'c' is replaced by 'Z'.

When all the 'a' are traversed and replaced by 'X', the machine traverses the right side to find if any untraversed 'b' or 'c' is left or not. If not, the machine gets Y and Z and at last a blank. Upon getting the blank symbol, the machine halts. The transitional functions are

$\delta(q_1, a) \rightarrow (q_2, X, R)$ // 'a' is traversed

$\delta(q_2, a) \rightarrow (q_2, a, R)$ // the remaining 'a's are traversed

$\delta(q_2, b) \rightarrow (q_3, Y, R)$ // 'b' is traversed

$\delta(q_3, b) \rightarrow (q_3, b, R)$ // the remaining 'b's are traversed

$\delta(q_3, c) \rightarrow (q_4, Z, L)$ // 'c' is traversed

$\delta(q_4, b) \rightarrow (q_4, b, L)$ // the remaining 'b's are traversed from right to left

$\delta(q_4, Y) \rightarrow (q_4, Y, L)$ // Y, replacement for 'b' is traversed

$\delta(q_4, a) \rightarrow (q_4, a, L)$ // the remaining 'a's are traversed from right to left

$\delta(q_4, X) \rightarrow (q_1, X, R)$ // rightmost X, replacement of 'a' is traversed

$\delta(q_2, Y) \rightarrow (q_2, Y, R)$ // this is used from the second time onwards

$\delta(q_3, Z) \rightarrow (q_3, Z, R)$ // this is used from the second time onwards

$\delta(q_4, Z) \rightarrow (q_4, Z, L)$ // this is used from the second time onwards

$\delta(q_1, Y) \rightarrow (q_5, Y, R)$ // this is used when all the 'a' are traversed

$\delta(q_5, Y) \rightarrow (q_5, Y, R)$ // this is used when all the 'a' are traversed and the machine is traversing right

$\delta(q_5, Z) \rightarrow (q_6, Z, R)$ // this is used when all the 'a' and 'b' are traversed.

$\delta(q_6, Z) \rightarrow (q_6, Z, R)$ // this is used when all the 'a' and 'b are traversed and the machine is traversing right to search for the remaining 'c'

$\delta(q_6, B) \rightarrow (q_f, B, H)$ // this is used when all the 'a', 'b', and 'c' are traversed

Upon executing the last transitional function, the machine halts.

20. Design a TM for a set of all strings with equal number of 'a' and 'b'.

Solution: The machine is designed as follows.

The machine first finds an 'a', replaces this by X and moves left. When it gets blank, it again moves right to search for 'b' and replaces it by Y. Again, it traverse left to search for 'a'. The process continues till the right side blank symbol is traversed in state q_0.

The transitional functions are

$\delta(q_0, a) \rightarrow (q_1, X, L)$ // search 'a' and traverse left

$\delta(q_0, b) \rightarrow (q_0, b, R)$

$\delta(q_0, X) \rightarrow (q_0, X, R)$

$\delta(q_0, Y) \rightarrow (q_0, Y, R)$

$\delta(q_1, B) \rightarrow (q_2, B, R)$ // traverse right in search of ‘b’

$\delta(q_1, X) \rightarrow (q_1, X, R)$

$\delta(q_1, Y) \rightarrow (q_1, Y, L)$

$\delta(q_1, b) \rightarrow (q_1, b, L)$

$\delta(q_2, a) \rightarrow (q_2, a, R)$

$\delta(q_2, X) \rightarrow (q_2, X, R)$

$\delta(q_2, Y) \rightarrow (q_2, Y, R)$

$\delta(q_2, b) \rightarrow (q_3, Y, L)$

$\delta(q_3, a) \rightarrow (q_3, a, L)$

$\delta(q_3, X) \rightarrow (q_3, X, L)$

$\delta(q_3, Y) \rightarrow (q_3, Y, L)$

$\delta(q_3, B) \rightarrow (q_0, B, R)$ // traverse right in search of
‘a’

$\delta(q_0, B) \rightarrow (q_f, B, H)$

21. Design a TM to accept the language $L = a^n b^{2n}$, $n > 0$.

Solution: The number of ‘b’ is equal to twice the number of ‘a’. In the language, all ‘a’ appear before ‘b’. The TM is designed as follows.

Replace one ‘a’ by X and traverse right to search for the end of the string. After getting a ‘B’, traverse left and replace two ‘b’s by Y. In the state q_0, if the machine gets Y as input it halts. The transitional functions are

$\delta(q_0, a) \rightarrow (q_1, X, R)$

$\delta(q_1, a) \rightarrow (q_1, a, R)$

$\delta(q_1, b) \rightarrow (q_1, b, R)$

$\delta(q_1, B) \rightarrow (q_2, B, L)$

$\delta(q_1, Y) \rightarrow (q_2, Y, L)$ // second traverse onwards

$\delta(q_2, b) \rightarrow (q_3, Y, L)$

$\delta(q_3, b) \rightarrow (q_4, Y, L)$

$\delta(q_4, b) \rightarrow (q_4, b, L)$

$\delta(q_4, a) \rightarrow (q_4, a, L)$

$\delta(q_4, X) \rightarrow (q_0, X, R)$

$\delta(q_0, Y) \rightarrow (q_f, Y, H)$.