MODULE-3

Context free grammar (CFG):-

A CFG 'G' is having four tuples $G=(V,T,P,S)$ where V represents set of variables that is also called as non-terminals and it is represented by capital letters. T represents set of terminal symbols and it is represented by lower case letters. S is a start symbol (one 4 variable). P is a set of productions or rules that represent the recursive definition of a language. Each production consists of

i) A variable that is being defined by production, this variable is often called the head of the production.

ii) The production symbol is single arrow ($\longrightarrow$)

iii) A string of 0 or more terminals followed by variables: This string is called the body of the production.

Example:- Valid productions:-

$A \longrightarrow aB$

$A \longrightarrow aTbabbS$

Invalid productions

$aB \longrightarrow A$

$AB \longrightarrow a$

Production:-

$V=\{B,A\}$ $B \longrightarrow A$

$T=\{a\}$ $A \rightarrow a$

$S=B$

$$\boxed{\begin{array}{l} B \Rightarrow A \\ \Rightarrow a \end{array}}$$

1. Obtain a grammar to generate string consisting of any number of a's.

$L=\{\epsilon, a, aa, aaa, \ldots\}$

$S \longrightarrow \epsilon \mid aS$

1) $S \Rightarrow aS$
$\Rightarrow a.\epsilon$

2) $S \Rightarrow aS$
$\Rightarrow aaS$
$\Rightarrow aa\epsilon$

3) $S \Rightarrow aS$
$\Rightarrow aaS$
$\Rightarrow aaaS$
$\Rightarrow aaa\epsilon$

2. Obtain a grammar to generate string consisting of atleast one a.

$S \rightarrow a \mid aS$

3. Obtain a grammar to generate string consisting of any a's or b's.

$S \rightarrow \epsilon \mid aS \mid bS$

aba

$S \Rightarrow aS$
$\Rightarrow abS$
$\Rightarrow abaS$
$\Rightarrow aba\epsilon$

4. Obtain a grammar to generate string consisting of atleast two a's

$S \rightarrow aa \mid aS$

$S \Rightarrow aS$
$\Rightarrow aaa$

5. Obtain a grammar to generate string consisting of even number of a's

$A \rightarrow \epsilon \mid aaA$

$A \Rightarrow aaA$
$\Rightarrow aa\epsilon$

6. Obtain a grammar to generate string consisting of multiple of 3 a's

$B \rightarrow \epsilon \mid aaaB$

$B \Rightarrow aaaB$
$\Rightarrow aaa\epsilon$

$B \Rightarrow aaaB$
$\Rightarrow aaaaaaB$
$\Rightarrow aaaaaa\epsilon$

7. Obtain a grammar to generate string consisting of a's and b's such that string length is multiple of 3.

$S \rightarrow \epsilon \mid AAAS$
$A \rightarrow a \mid b$

abbaab

$S \Rightarrow AAAS$
$\Rightarrow aAAS$
$\Rightarrow abAS$
$\Rightarrow abbS$
$\Rightarrow abbAAAS$
$\Rightarrow abbaAAS$
$\Rightarrow abbaaAS$
$\Rightarrow abbaabS$
$\Rightarrow abbaab\epsilon$

8. Obtain a string consisting of a's and b's with atleast one a or one b.

$A \rightarrow a \mid b \mid aS \mid bS$

9. Obtain a grammar to accept the following language $L=\{W \in a^*: |W| \bmod 3 = 0\}$

$$A \rightarrow \epsilon \mid aaaA$$

10. Obtain a grammar to accept the following language $L=\{W \in a^*: |W| \bmod 3 \neq$

$$A \rightarrow a \mid aa \mid aaaA$$

11. Obtain a grammar to generate strings of a's and b's having a substring ab.

$$S \rightarrow AabA$$
$$A \rightarrow \epsilon \mid aA \mid bA$$

12. Obtain a grammar to generate strings of a's and b's beginning with aba

$$S \rightarrow abaA$$
$$A \rightarrow \epsilon \mid aA \mid bA$$

13. Obtain a grammar to generate strings of a's and b's ending with bab

$$S \rightarrow Abab$$
$$A \rightarrow \epsilon \mid aA \mid bA$$

14. Obtain a grammar to accept the following language $L=\{W \in \{a,b\}^*: \#_a(W) \bmod 2 = 0\}$

$$S \rightarrow BaBaBS \mid B$$
$$B \rightarrow \epsilon \mid bB$$

15. Obtain a grammar to generate the following language $L=\{a^n b^n \mid n \geq 0\}$

$$S \rightarrow \epsilon \mid aSb$$

16. Obtain a grammar to generate the following language $L=\{a^n b^n \mid n \geq 1\}$

$$S \rightarrow ab \mid aSb$$

17. Obtain a grammar to generate the following language $L=\{a^{n+1} b^n \mid n \geq 0\}$

$$S \rightarrow a \mid aSb$$

18. Obtain a grammar to generate the following language $L=\{a^n b^{n+2}: n \geq 0\}$ $S \rightarrow bb \mid aSb$

19. Obtain a grammar to generate the following language

$L = \{a^n b^{2n} : n \geq 0\}$

$S \rightarrow \epsilon \mid aSbb$

20. Obtain a grammar to generate the following language.

$L = \{a^{2n} b^n : n \geq 1\}$

$S \rightarrow aab \mid aaSb$

21. Obtain a grammar to generate the language $L = \{0^m 1^m 2^n : m \geq 1, n \geq 0\}$

$S \rightarrow AB$

$A \rightarrow 01 \mid 0A1$

$B \rightarrow \epsilon \mid 2B$

22. Obtain a grammar to generate the language $L = \{W \in \{a,b\}^* : WW^R\}$

$S \rightarrow \epsilon \mid aSa \mid bSb$

babbab

$S \Rightarrow bSb$

$\Rightarrow baSab$

$\Rightarrow babSbab$

$\Rightarrow bab\epsilon bab$

23. Obtain a grammar to generate $L = \{W \in \{a,b\}^* :$ number of a's in $W =$ number of b's in $W\}$.

1) $S \rightarrow \epsilon \mid aSb \mid bSa \mid SS$

2) $S \rightarrow \epsilon \mid aSb \mid bSa \mid abS \mid baS$

3) $S \rightarrow \epsilon \mid aSbS \mid bSaS$

24. Obtain a grammar to generate $L = \{W \in \{a\}^* : |W| \bmod 3 \neq |W| \bmod 2\}$

$L = \{2, 3, 4, 5, \cancel{6}, \cancel{7},$
$8, 9, 10, 11, \cancel{12}, \cancel{13},$
$14, 15, 16, 17, \cancel{18}, \cancel{19}\}$

$S \rightarrow aa \mid aaa \mid aaaa \mid aaaaa \mid aaaaaaS$

25. Obtain a grammar to generate $L = \{W \in \{a\}^* : |W| \bmod 3 \geq |W| \bmod 2\}$

$L = \{0, 1, 2, 4, 5, 6, 7, 8, 10, 11, 12, 13, 14, 16, 17, 18, 19, 20, 22\}$

$S \rightarrow \epsilon \mid a \mid aa \mid aaaa \mid aaaaa \mid aaaaaaS$

26. Obtain a grammar to generate $L = \{W \in \{a\}^* : |W| \bmod 3 \leq |W| \bmod 2\}$

$L = \{0, 1, 3, \cancel{4}, 6, 7, 9, 12, 13, 15, 18\}$

$S \rightarrow \epsilon \mid a \mid aaa \mid aaaaaaS$

27. Obtain a grammar to generate $L = \{W \in \{a, b\}^* : |W| \bmod 3 \leq |W| \bmod 2\}$

$S \rightarrow \epsilon \mid A \mid AAA \mid AAAAAAS$

$A \rightarrow a \mid b$

28. Obtain a grammar to generate $L = \{a^{n+2} b^m \mid n \geq 0 \text{ and } m > n\}$

$n = 0$ then $m \geq 1$

$n=0 \quad aabb^*$

$n=1 \quad aaabbb^*$

$n=2 \quad aaaabbbb^*$

$n=3 \quad aaaaabbbbb^*$

$S \rightarrow aAB$

$A \rightarrow ab \mid aAb$

$B \rightarrow \epsilon \mid bB$

29. $L = \{a^n b^m \mid n \geq 0 \text{ and } m > n\}$

$n = 0$ then $m \geq 1$

$n=0, \quad bb^*$

$n=1 \quad abbb^*$

$n=2 \quad aabbbb^*$

$n=3 \quad aaabbbbb^*$

$S \rightarrow AB$

$A \rightarrow \epsilon \mid aAb$

$B \rightarrow b \mid bB$

30. $L = \{a^n b^m c^k \mid n + 2m = k, n \geq 0, m \geq n\}$

$S \rightarrow aSc \mid A$

$A \rightarrow \epsilon \mid bAcc$

$a^n b^m c^{n+2m}$

$a^n b^m c^n c^{2m}$

$a^n \underbrace{b^m c^{2m}}_{A} c^n$

31. Obtain a grammar to generate $L = \{0^i 1^j \mid i \neq j, i \geq 0, j \geq 0\}$

$i=0$ then $j \geq 1$

$i=0 \quad 11^*$

$i=1 \quad 0111^*$

$i=2 \quad 001111^*$

$i=3 \quad 00011111^*$

$j=0$ then $i \geq 1$

$i=1, j=0 \quad 0$

$i=2, j=1 \quad 0011^*$

$i=3, j=2 \quad 000111^*$

$S \rightarrow BA \mid AC$

$A \rightarrow \epsilon \mid 0A1$

$B \rightarrow 0 \mid 0B$

$C \rightarrow 1 \mid 1C$

Derivations and Parse trees:-

There are two types of derivations

i) left most derivation (LMD) from left to right

ii) Right most derivation (RMD) from right to left

1. Obtain LMD, RMD and Parse tree the string $(a101+b1)*(a1+b)$ from the following grammar

$E \rightarrow E*E \mid E+E \mid (E) \mid I$

$I \rightarrow a \mid b \mid Ia \mid Ib \mid I0 \mid I1$

$E \underset{LMD}{\Rightarrow} E*E$

$\Rightarrow (E)*E$

$\Rightarrow (E+E)*E$

$\Rightarrow (I+E)*E$

$\Rightarrow (I1+E)*E$

$\Rightarrow (I01+E)*E$

$\Rightarrow (I101+E)*E$

$\Rightarrow (a101+E)*E$

$\Rightarrow (a101+I)*E$

$\Rightarrow (a101+I1)*E$

$\Rightarrow (a101+b1)*E$

$\Rightarrow (a101+b1)*(E)$

$\Rightarrow (a101+b1)*(E+E)$

$\Rightarrow (a101+b1)*(I+E)$

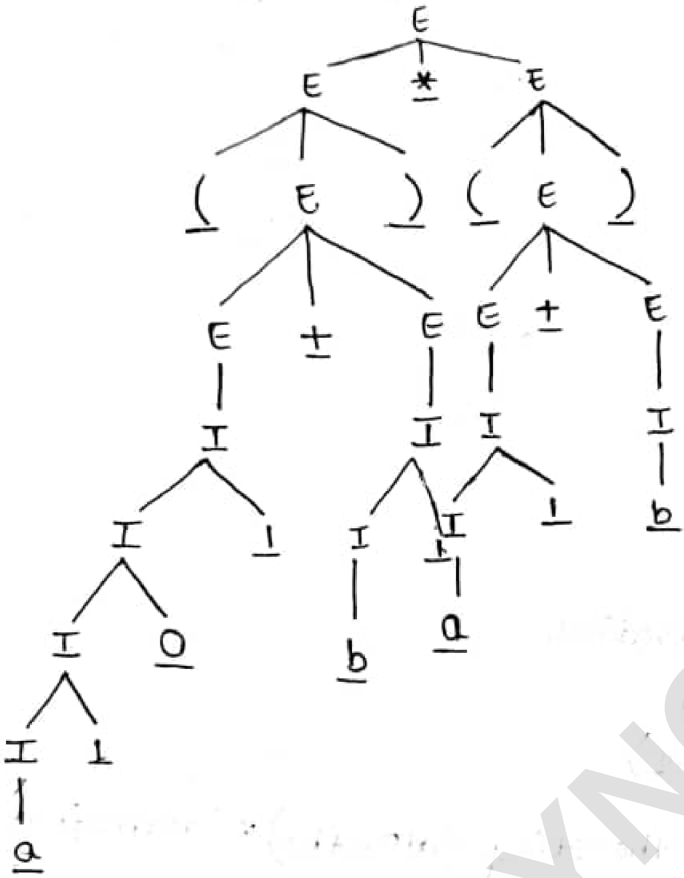
$\Rightarrow (a101+b1)*(I1+E)$

$\Rightarrow (a101+b1)*(a1+E)$

$\Rightarrow (a101+b1)*(a1+I)$

$\Rightarrow (a101+b1)*(a1+b)$

Parse tree:-



$E \underset{RMD}{\Rightarrow} E*E$

$\Rightarrow E*(E)$

$\Rightarrow E*(E+E)$

$\Rightarrow E*(E+I)$

$\Rightarrow E*(E+b)$

$\Rightarrow E*(I+b)$

$\Rightarrow E*(I1+b)$

$\Rightarrow E*(a1+b)$

$\Rightarrow (E)*(a1+b)$

$\Rightarrow (E+E)*(a1+b)$

$\Rightarrow (E+I)*(a1+b)$

$\Rightarrow (E+I1)*(a1+b)$

$\Rightarrow (E+b1)*(a1+b)$

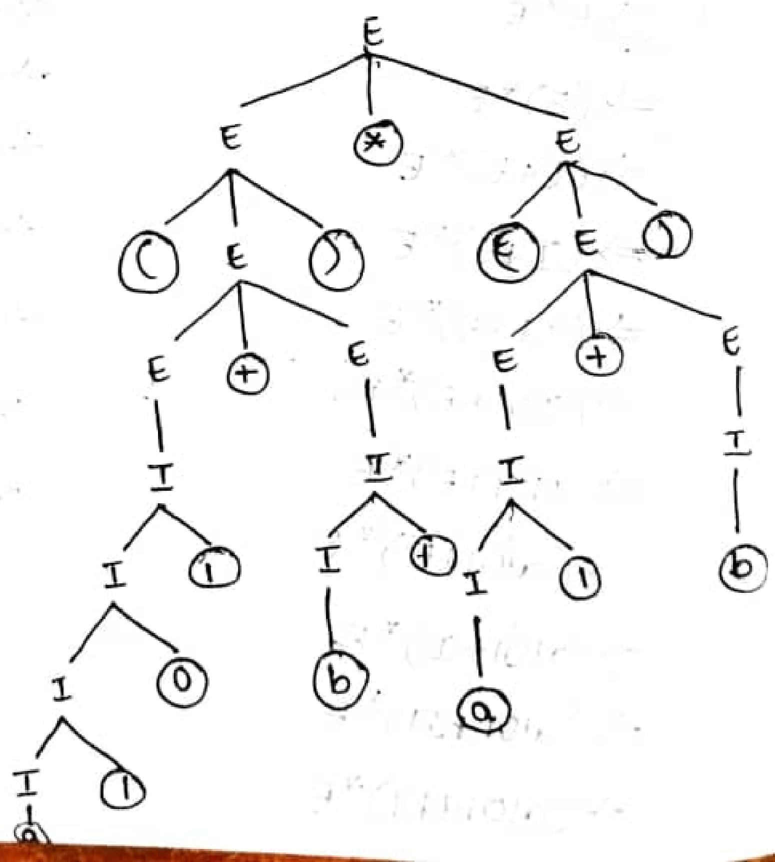
$\Rightarrow (I+b1)*(a1+b)$

$\Rightarrow (I1+b1)*(a1+b)$

$\Rightarrow (I01+b1)*(a1+b)$

$\Rightarrow (I101+b1)*(a1+b)$

$\Rightarrow (a101+b1)*(a1+b)$



2. Obtain LMD, RMD, Parse tree for the string $+-*xyxy$ from the grammar

$E \rightarrow +EE \mid -EE \mid *EE \mid I$

$I \rightarrow x \mid y$

$E \underset{LMD}{\Longrightarrow} +EE$

$\Rightarrow +-EEE$

$\Rightarrow +-*EEEE$

$\Rightarrow +-*IEEE$

$\Rightarrow +-*xEEE$

$\Rightarrow +-*xIEE$

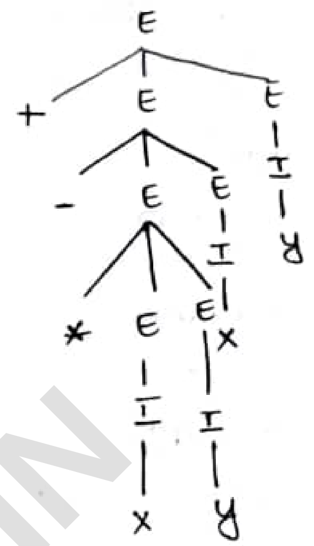
$\Rightarrow +-*xyEE$

$\Rightarrow +-*xyIE$

$\Rightarrow +-*xyxE$

$\Rightarrow +-*xyxI$

$\Rightarrow +-*xyxy$



$E \underset{RMD}{\Longrightarrow} +EE$

$\Rightarrow +EI$

$\Rightarrow +Ey$

$\Rightarrow +-EEy$

$\Rightarrow +-EIy$

$\Rightarrow +-Exy$

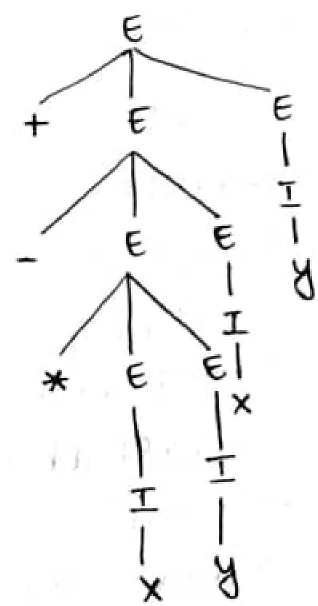
$\Rightarrow +-*EExy$

$\Rightarrow +-*EIxy$

$\Rightarrow +-*Eyxy$

$\Rightarrow +-*Iyxy$

$\Rightarrow +-*xyxy$



3. Consider the grammar

$S \rightarrow aB \mid bA$

$A \rightarrow aS \mid bAA \mid a$

$B \rightarrow bS \mid aBB \mid b$

Obtain LMD, RMD, Parse trees for the string aaabbabbba

$S \underset{LMD}{\Rightarrow} aB$

$\Rightarrow aaBB$

$\Rightarrow aaaBBB$

$\Rightarrow aaabSBB$

$\Rightarrow aaabbABB$

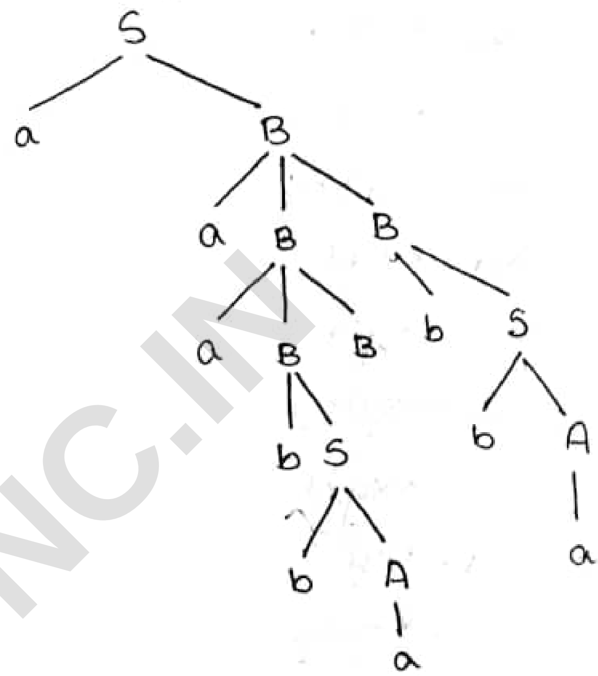
$\Rightarrow aaabbaBB$

$\Rightarrow aaabbabB$

$\Rightarrow aaabbabbS$

$\Rightarrow aaabbabbbA$

$\Rightarrow aaabbabbba$



$S \underset{RMD}{\Rightarrow} aB$

$\Rightarrow aaBB$

$\Rightarrow aaBbS$

$\Rightarrow aaBbbA$

$\Rightarrow aaBbba$

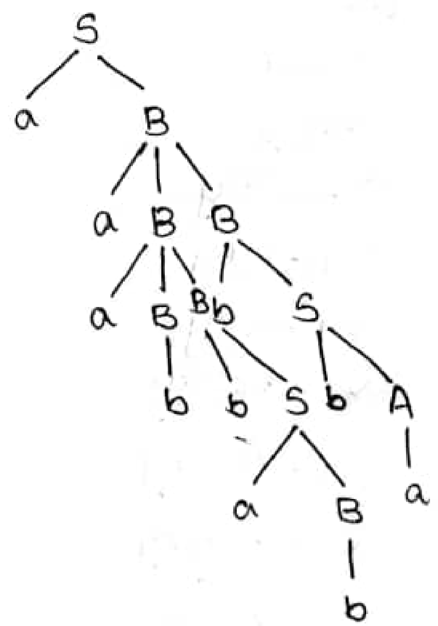
$\Rightarrow aaaBBbba$

$\Rightarrow aaaBbSbba$

$\Rightarrow aaaBbaBbba$

$\Rightarrow aaaBbabbba$

$\Rightarrow aaabbabbba$



Pumping Lemma for Context free Language:-

Theorem:- Let L be a CFL, then there exists a constant n such that if z in any string in L such that |z| is atleast n then we can write $z = uvwxy$ subject to the following condition.

1) $|vwx| \leq n$ (The middle portion is not too long)

2) $vx \neq \epsilon$ i.e, $|vx| \geq 1$ (Atleast one of the strings we pump must not be empty)

3) For all $i \geq 0$, uv^iwx^iy is in L (that is two strings v and x may be pumped any number of times).

Show that $L = \{a^n b^n c^n \mid n \geq 0\}$ is not context free.

Closure properties of CFL's:-

1) CFLs are closed under union, concatenation and star

i) Union of CFLs is CFL:-

Let L1 and L2 are two CFLs generated by CFGs.

$G_1 = (V_1, T_1, P_1, S_1)$

$G_2 = (V_2, T_2, P_2, S_2)$

Let us consider the language L3 generated by the grammar

$G_3 = (V_1 \cup V_2 \cup S_3, T_1 \cup T_2, P_3, S_3)$

where S3 is the start symbol from G3.

$S_3 \notin (V_1 \cup V_2)$

$P_3 = P_1 \cup P_2 \cup \{S_3 \rightarrow S_1 | S_2\}$

$L3 = L1 \cup L2$

G3 in context free and the language generated by this grammar is context free. If we assume $W \in L$, then the possible derivation from S3 is

$S3 \Rightarrow S1$

$\Rightarrow W$

If we assume $W \in L2$

$S3 \Rightarrow S2$

$\Rightarrow W$

So if $W \in L3$ one of the derivation

$S3 \Rightarrow S1 \Rightarrow W$

or $S3 \Rightarrow S2 \Rightarrow W$ is possible

So, $L3 = L1 \cup L2$

Thus it is proved that context free languages are closed under union.

ii) Concatenation of two CFLs is CFL:-

Let us consider the language L4 generated by the grammar

$G4 = (V1 \cup V2 \cup S4, T1 \cup T2, P4, S4)$

$S4 \notin (V1 \cup V2)$

$P4 = P1 \cup P2 \cup \{S4 \rightarrow S1S2\}$

It is clear from this that the grammar G4 is context free and language generated by this grammar is context free and so

$L3 = L1L2$

Thus, it is proved that CFL are closed under concatenation.

iii) CFLs are closed under star closure:-

Now, let us consider the language L5 generated by grammar

$G5 = (V1 \cup S5, T1, P5, S5)$

S5 is start symbol of G5

$P5 = P1 \cup \{S5 \rightarrow S1S5 | \epsilon$

$L5 = L5^*$

It is proved that CFL are closed under star-closure

iv) CFLs are not closed under intersection:-

CFLs are not closed under intersection. If L1 and L2 are CFL, it is not always true that $L_1 \cap L2$ and CFL.

Consider two languages

$L1 = \{a^n b^n c^m \mid n \geq 0, m \geq 0\}$ and

$L2 = \{a^n b^m c^m \mid n \geq 0, m \geq 0\}$

The two languages are context free, as we can easily obtain the corresponding CFG.

$S \rightarrow S1S2$

$S1 \rightarrow aS1b | \epsilon$

$S2 \rightarrow cS2 | \epsilon$

and $S \rightarrow aS | S1$

$S1 \rightarrow bS_1C | \epsilon$

$L1 \cap L2 = \{a^n b^n c^n \mid n \geq 0 \mid n \geq 0\}$

We have already proved earlier that this language is not CFL.

So, it is not closed under intersection.

1. Show that $L = \{a^n b^n c^n \mid n \geq 0\}$ is context free language

$n = 3$

$W = \underbrace{aaa}_{u}\underbrace{b}_{v}\underbrace{b}_{w}\underbrace{b}_{x}\underbrace{ccc}_{y}$

$|W| \geq n$

$9 > 3$

i) $|vwx| \leq n$

$3 = 3$

ii) $vx \neq \epsilon$

$bb = vx$

iii) $uv^i wx^i y \in L$

$i=0$ $uv^0wx^0y = aaabccc \notin L$

$i=1$ $uv^1wx^1y = aaabbbccc \in L$

$i \geq 2$ $uv^2wx^2y = aaabbbbbccc \notin L$

For $i=0$ and ≥ 2 the assumption is false

2. Show that $L = \{a^p b^q \mid p = q^2\}$ is not context free language

Let $p = 9$, $q = 3$

$W = \underbrace{aaaaaa}_{u}\underbrace{a}_{v}\underbrace{a}_{w}\underbrace{a}_{x}\underbrace{bbb}_{y}$

i) $|vwx| \leq n$

$3 = 3$

ii) $vx \neq \epsilon$

$vx = aa$

iii) $\forall\ i \geq 0$ $uv^iwx^iy \in L$

$i=0$ $uv^0wx^0y = aaaa\epsilon a\epsilon abbb \notin L$

$i=1$ $uv^1wx^1y = aaaaaaaaabbb \in L$

$i \geq 2$ $uv^2wx^2y = aaaaaaaaaaabbb \notin L$

For $i=0$ and ≥ 2 the assumption is false.

Ambiguity of a Grammar:-

A CFG $G=\{V,T,P,S\}$ is ambiguous if there is atleast one string W for which we can find two different parse trees each with root label S produce W.

→ If each string has atmost one parse tree in the grammar, then the grammar is unambigiuous.

1. Consider the grammar.

$E \rightarrow E+E \mid E*E \mid I$

$I \rightarrow a$

Check whether it is ambiguious

$E \underset{LMD}{\Rightarrow} E+E$

$\Rightarrow E*E+E$

$\Rightarrow I*E+E$

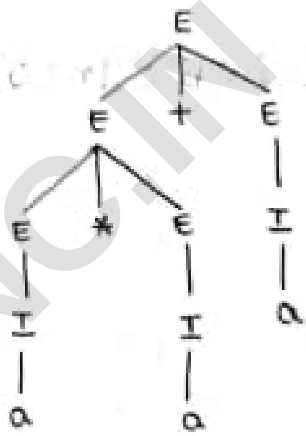
$\Rightarrow a*E+E$

$\Rightarrow a*I+E$

$\Rightarrow a*a+E$

$\Rightarrow a*a+I$

$\Rightarrow a*a+a$



$E \underset{LMD}{\Rightarrow} E*E$

$\Rightarrow I*E$

$\Rightarrow a*E$

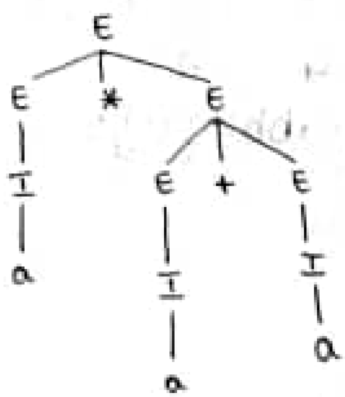
$\Rightarrow a*E+E$

$\Rightarrow a*I+E$

$\Rightarrow a*a+E$

$\Rightarrow a*a+I$

$\Rightarrow a*a+a$



For the same string $a*a+a$, we get two different parse trees, so the grammar is ambiguious.

2. Check whether the given grammar is ambiguious or not.

$$S \rightarrow aS \mid X$$
$$X \rightarrow aX \mid a$$

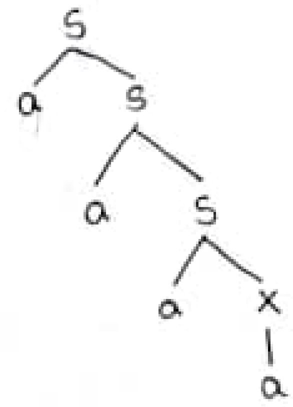
$S \underset{LMD}{\Longrightarrow} aS$

$\Longrightarrow aaS$

$\Longrightarrow aaX$

$\Longrightarrow aaaX$

$\Longrightarrow aaaa$



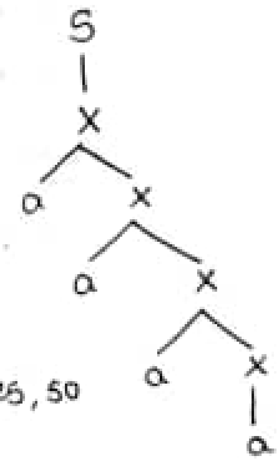
$S \underset{LMD}{\Longrightarrow} X$

$\Longrightarrow aX$

$\Longrightarrow aaX$

$\Longrightarrow aaaX$

$\Longrightarrow aaaa$



For the same string, we get two different parse trees, so the grammar is ambiguious.

3. Check whether the given grammar is ambiguious or not

$$S \rightarrow aSbS \mid bSaS \mid \epsilon$$

$S \underset{LMD}{\Longrightarrow} aSbS$

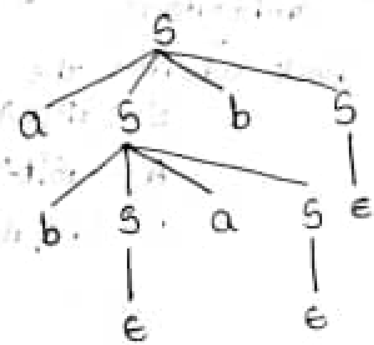
$\Rightarrow abSaSbS$

$\Rightarrow ab\epsilon aSbS$

$\Rightarrow ab\epsilon a\epsilon bS$

$\Rightarrow ab\epsilon a\epsilon b\epsilon$

$\Rightarrow abab$



$S \underset{LMD}{\Longrightarrow} aSbS$

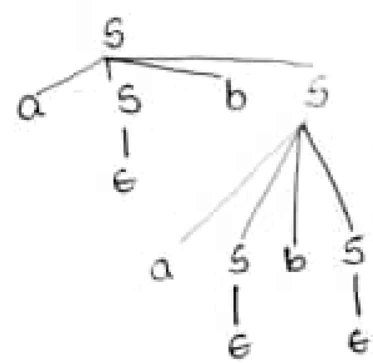
$\Rightarrow a\epsilon bS$

$\Rightarrow a\epsilon baSbS$

$\Rightarrow a\epsilon ba\epsilon bS$

$\Rightarrow a\epsilon ba\epsilon b\epsilon$

$\Rightarrow abab$

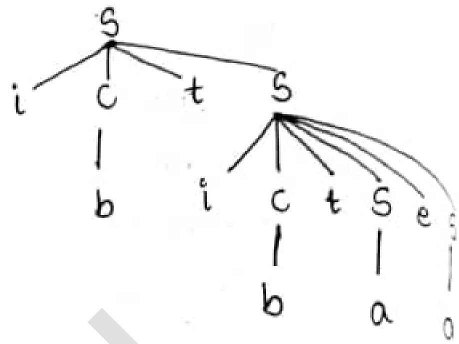


For the same string, we get two different parse trees, so the grammar is ambiguous.

4. Check whether the given grammar is ambiguous or not

$S \rightarrow iCtS \mid iCtSeS \mid a$

$C \rightarrow b$



$S \underset{LDM}{\Rightarrow} iCtS$

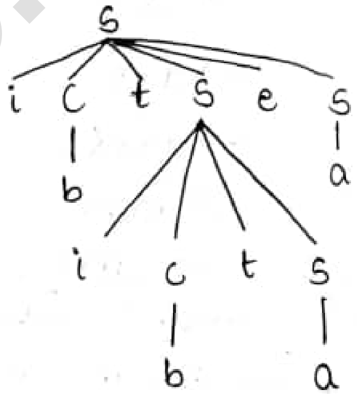
$\Rightarrow ibtS$

$\Rightarrow ibtiCtSeS$

$\Rightarrow ibtibtSeS$

$\Rightarrow ibtibtaeS$

$\Rightarrow ibtibtaea$



$S \underset{LDM}{\Rightarrow} iCtSeS$

$\Rightarrow ibtSeS$

$\Rightarrow ibtiCtSeS$

$\Rightarrow ibtibtSeS$

$\Rightarrow ibtibtaeS$

$\Rightarrow ibtibtaea$

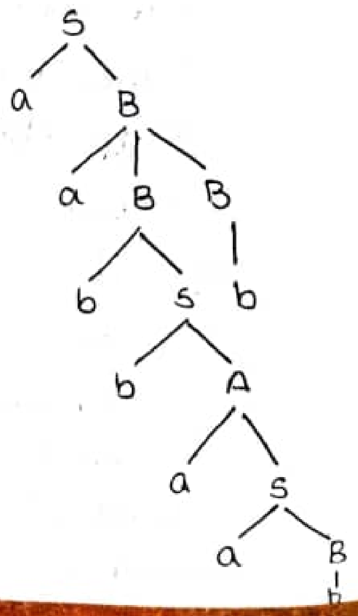
For the same thing, we get two different parse trees, so the grammar is ambiguous.

5. Check whether the given grammar is ambiguous or not

$S \rightarrow aB \mid bA$

$A \rightarrow aS \mid bAA \mid a$

$B \rightarrow bS \mid aBB \mid b$



$S \underset{LMD}{\Rightarrow} aB$

$\Rightarrow aaBB$

$\Rightarrow aabSB$

$\Rightarrow aabbAB$

$\Rightarrow aabbaaBB$

$\Rightarrow aabbaabb$

$S \underset{LMD}{\Rightarrow} aB$

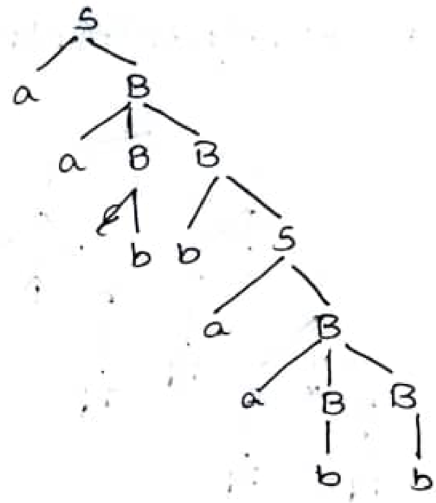
$\Rightarrow aaBB$

$\Rightarrow aabbS$

$\Rightarrow aabbaB$

$\Rightarrow aabbaaBB$

$\Rightarrow aabbaabb$



Elimination of Ambiguity

There are mainly two reasons to get ambiguious grammar

1) Associativity
2) Precedence

Associativity can be removed by using recurssion.

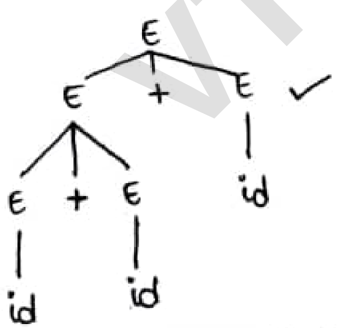
$+, -, *, / \Rightarrow$ Left associate

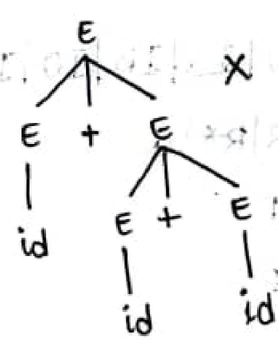
$\uparrow \Rightarrow$ Right associate

Precedence can be removed by levels i.e, highest precedence operation should be in bottom level.

Consider the grammar

i) $E \rightarrow E+E \mid E*E \mid id$



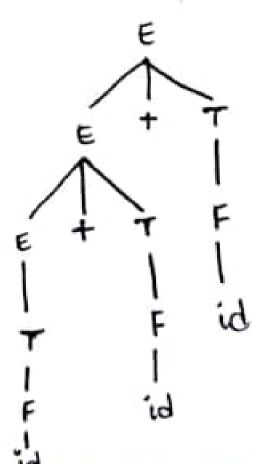


Unambiguious grammar

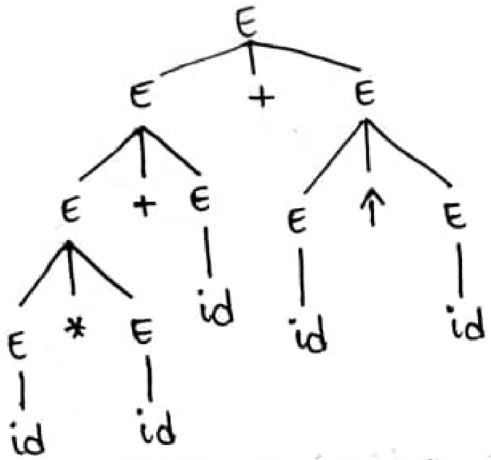
$E \rightarrow E+T \mid T$

$T \rightarrow T*F \mid F$

$F \rightarrow id$



ii) $E \rightarrow E+E \mid E*E \mid E\uparrow E \mid id$



id* id+id+id↑id

$E \rightarrow E+T \mid T$
$T \rightarrow T*F \mid F$
$F \rightarrow G*F \mid G$
$G \rightarrow id$

iii) $E \rightarrow E*E \mid E+E \mid (E) \mid I$
$I \rightarrow a \mid b \mid Ia \mid Ib \mid I0 + I1$

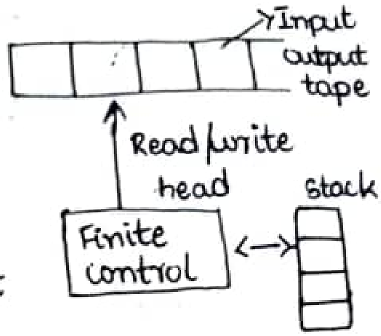
$E \rightarrow E+T \mid T$
$T \rightarrow T*F \mid F$
$F \rightarrow (E) \mid I$
$I \rightarrow a \mid b \mid Ia \mid Ib \mid I0 \mid I1$

iv) $R \rightarrow R+R \mid R.R \mid R* \mid a$

$R \rightarrow R+T \mid T$
$T \rightarrow T.F \mid F$
$F \rightarrow F* \mid a$

PDA [Push Down Automata]:-

Definition:-



PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ where Q, Σ, q_0, F are same as DFA (or) NFA.

Z_0 is the stack start symbol

Γ (big gamma) is a finite stack alphabet

δ is a transition function that takes three arguments a input and returns two arguments.

$$\delta(q, a, x) = (p, \gamma)$$

q → current state; a → Input symbol; x → stack symbol; p → next state; γ → string of stack symbol that replaces x at the top of the stack.

1. Consider the language $L = \{a^n b^n \mid n \geq 1\}$. Obtain a PDA to accept the language by final state and show the Instantaneous Description (ID) moves for the string aaabbb from the same PDA as above and draw the transition diagram for the same.

Deterministic PDA

1) $\delta(q_0, a, Z_0) = (q_0, aZ_0)$ } Push operation

2) $\delta(q_0, a, a) = (q_0, aa)$

3) $\delta(q_0, b, a) = (q_1, \epsilon)$ } Pop operation

4) $\delta(q_1, b, a) = (q_1, \epsilon)$

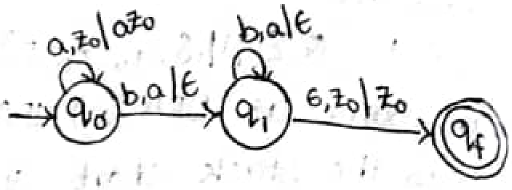
5) $\delta(q_1, \epsilon, Z_0) = (q_f, Z_0)$

ID moves:-

$$(q_0, aaabbb, Z_0) \vdash (q_0, aabbb, aZ_0)$$
$$\vdash (q_0, abbb, aaZ_0)$$
$$\vdash (q_0, bbb, aaaZ_0)$$
$$\vdash (q_1, bb, aaZ_0)$$
$$\vdash (q_1, b, aZ_0)$$

$\vdash (q_1, \epsilon, Z_0)$

$\vdash (q_f, \epsilon, Z_0)$



ii> aabbb

1> $\delta(q_0, a, Z_0) = (q_0, aZ_0)$

2> $\delta(q_0, a, a) = (q_0, aa)$

3> $\delta(q_0, b, a) = (q_1, \epsilon)$

4> $\delta(q_1, b, a) = (q_1, \epsilon)$

5> $\delta(q_1, b, Z_0) = (q_1, b)$

ID moves:-

$(q_0, aabbb, Z_0) \vdash (q_0, abbb, aZ_0)$

$\vdash (q_0, bbb, aaZ_0)$

$\vdash (q_1, bb, aZ_0)$

$\vdash (q_1, b, Z_0)$

String is not reaching to final state. So it is not accepted.

Languages of PDA:-

1> Acceptance by final state:-

Let $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ be a PDA then $L(P) = \{W \mid (q_0, W, Z_0) \vdash^* (q, \epsilon, \alpha)\}$

For some state q in F and any stack string α

2> Acceptance by empty stack:-

Let $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ be a PDA then $N(P) = \{W \mid (q_0, W, Z_0) \vdash^* (q, \epsilon, \epsilon)\}$

For any state q and stack content should be ϵ and the input string also ϵ.

2. Obtain a PDA for the string $L = \{WCW^R \mid W \in \{a,b\}^*\}$ by final state.

abbCbba

baaCaab

babCbab

abaCaba

$$\left.\begin{aligned} \delta(q_0, a, Z_0) &= (q_0, aZ_0) \\ \delta(q_0, b, Z_0) &= (q_0, bZ_0) \end{aligned}\right\} \text{Initialize}$$

$$\left.\begin{aligned} \delta(q_0, b, a) &= (q_0, ba) \\ \delta(q_0, a, b) &= (q_0, ab) \\ \delta(q_0, a, a) &= (q_0, aa) \\ \delta(q_0, b, b) &= (q_0, bb) \end{aligned}\right\} \text{Push}$$

Deterministic PDA

$$\left.\begin{aligned} \delta(q_0, C, b) &= (q_1, b) \\ \delta(q_0, C, a) &= (q_1, a) \end{aligned}\right\} \text{Shift}$$

$$\delta(q_0, C, Z_0) = (q_f, Z_0)$$

$$\left.\begin{aligned} \delta(q_1, b, b) &= (q_1, \epsilon) \\ \delta(q_1, a, a) &= (q_1, \epsilon) \end{aligned}\right\} \text{Pop}$$

$$\delta(q_1, \epsilon, Z_0) = (q_f, Z_0)$$

ID moves :-

$$\begin{aligned} (q_0, abbCbba, Z_0) &\vdash (q_0, bbCbba, aZ_0) \\ &\vdash (q_0, bCbba, baZ_0) \\ &\vdash (q_0, Cbba, bbaZ_0) \\ &\vdash (q_1, bba, bbaZ_0) \\ &\vdash (q_1, ba, baZ_0) \\ &\vdash (q_1, a, aZ_0) \\ &\vdash (q_f, \epsilon, Z_0) \end{aligned}$$

q_0 q_1 q_f

abCbb

$(q_0, abCbb, Z_0) \vdash (q_0, bCbb, aZ_0)$

$\vdash (q_0, Cbb, baZ_0)$

$\vdash (q_0, bb, baZ_0)$

$\vdash (q_1, b, aZ_0)$

The stack is not empty. So the string is not accepted.

3. Obtain a PDA for a language $L = \{W \in \{a,b\}^* : N_a(W) = N_b(W)\}$ by final state and draw the transition diagram and show the ID moves for string ababab, babaa.

$\delta(q_0, a, Z_0) = (q_0, aZ_0)$

$\delta(q_0, b, Z_0) = (q_0, bZ_0)$

$\delta(q_0, a, a) = (q_0, aa)$

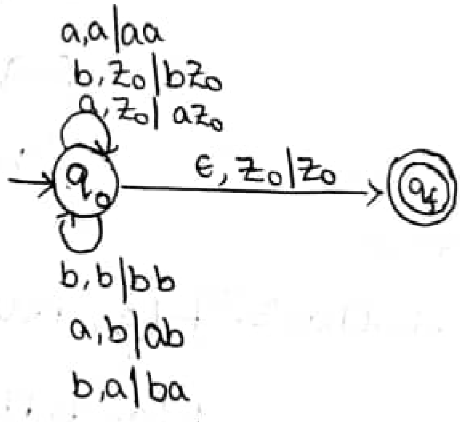
$\delta(q_0, b, b) = (q_0, bb)$

$\delta(q_0, a, b) = (q_0, \epsilon)$

$\delta(q_0, b, a) = (q_0, \epsilon)$

$\delta(q_0, \epsilon, Z_0) = (q_f, Z_0)$

Non-deterministic PDA



ababab

$(q_0, ababab, Z_0) \vdash (q_0, babab, aZ_0)$

$\vdash (q_0, abab, Z_0)$

$\vdash (q_0, bab, aZ_0)$

$\vdash (q_0, ab, Z_0)$

$\vdash (q_0, b, aZ_0)$

$\vdash (q_f, \epsilon, Z_0)$

The stack is empty. Therefore, it is accepted.

babaa

$(q_0, babaa, Z_0) \vdash (q_0, abaa, bZ_0)$

$\vdash (q_0, baa, Z_0)$

$\vdash (q_0, aa, bZ_0)$

$\vdash (q_0, a, Z_0)$

The stack is not empty. Therefore, it is not accepted.

4. Obtain a PDA to accept the language $L = \{a^n b^{2n} \mid n \geq 1\}$. Draw the transition diagram for the same and show the ID moves ~~aabbbb~~, aabbbb and aabbb.

$\delta(q_0, a, Z_0) = (q_0, aaZ_0)$

$\delta(q_0, a, a) = (q_0, aaa)$

$\delta(q_0, b, a) = (q_1, \epsilon)$

$\delta(q_1, b, a) = (q_1, \epsilon)$

$\delta(q_1, \epsilon, Z_0) = (q_f, Z_0)$

ID moves:-

aabbbb

$(q_0, aabbbb, Z_0) \vdash (q_0, abbbb, aaZ_0)$

$\vdash (q_0, bbbb, aaaaZ_0)$

$\vdash (q_1, bbb, aaaZ_0)$

$\vdash (q_1, bb, aaZ_0)$

$\vdash (q_1, b, aZ_0)$

$\vdash (q_f, \epsilon, Z_0)$

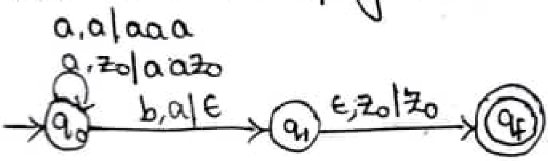
$(q_0, aabbb, z_0) \vdash (q_0, abbb, az_0)$

$\vdash (q_0, bbb, aaz_0)$

$\vdash (q_0, bb, az_0)$

$\vdash (q_0, b, z_0)$

The stack is not empty. So the string is not accepted



5. Obtain a PDA to accept the language $L = \{WW^R \mid W \in \{a,b\}^*\}$ by final state. Draw the transition diagram for the same and show the ID moves for the string aabbaa, ababa

$\delta(q_0, a, z_0) = (q_0, az_0)$
$\delta(q_0, b, z_0) = (q_0, bz_0)$
$\delta(q_0, \epsilon, z_0) = (q_0, z_0)$

Initialization

$\delta(q_0, a, a) = (q_0, aa)$
$\delta(q_0, a, b) = (q_0, ab)$
$\delta(q_0, b, b) = (q_0, bb)$
$\delta(q_0, b, a) = (q_0, ba)$

Push

$\delta(q_0, a, a) = (q_1, \epsilon)$
$\delta(q_0, b, b) = (q_1, \epsilon)$
$\delta(q_1, a, a) = (q_1, \epsilon)$
$\delta(q_1, b, b) = (q_1, \epsilon)$

Pop

$\delta(q_1, \epsilon, z_0) = (q_f, z_0)$

ID moves:-

aabbaa

$$(q_0, aabbaa, Z_0) \vdash (q_0, abbaa, aZ_0)$$
$$\vdash (q_0, bbaa, aaZ_0)$$
$$\vdash (q_0, baa, baaZ_0)$$
$$\vdash (q_1, aa, aaZ_0)$$
$$\vdash (q_1, a, aZ_0)$$
$$\vdash (q_f, \epsilon, Z_0)$$

The stack is empty. So the string is accepted.

ababa

$$(q_0, ababa, Z_0) \vdash (q_0, ababa, aZ_0)$$
$$\vdash (q_0, aba, baZ_0)$$
$$\vdash (q_0, ba, abaZ_0)$$
$$\vdash (q_0, a, babaZ_0)$$

The stack is not empty. So, the string is not accepted.

6. Obtain a PDA to accept a string of balanced paranthesis. The paranthesis to be considered are (,), [,].

$$\left.\begin{array}{l} \delta(q_0, (, Z_0) = (q_0, (Z_0) \\ \delta(q_0, [, Z_0) = (q_0, [Z_0) \end{array}\right\} \text{Initialization}$$

$$\left.\begin{array}{l} \delta(q_0,), () = (q_0, \epsilon) \\ \delta(q_0,], [) = (q_0, \epsilon) \end{array}\right\} \text{Pop}$$

$$\left.\begin{array}{l} \delta(q_0, (, () = (q_0, (() \\ \delta(q_0, [, [) = (q_0, [[) \\ \delta(q_0, (, [) = (q_0, ([) \\ \delta(q_0, [, () = (q_0, [() \end{array}\right\} \text{Push}$$

$$\delta(q_0, \epsilon, Z_0) = (q_f, Z_0)$$

ID moves - ([])()

$(q_0, ([])(), Z_0) \vdash (q_0, [])(), (Z_0)$

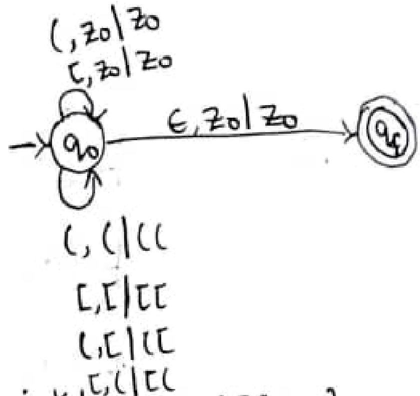
$\vdash (q_0,])(), [(Z_0)$

$\vdash (q_0,)(), (Z_0)$

$\vdash (q_0, (), Z_0)$

$\vdash (q_0,), (Z_0)$

$\vdash (q_f, \epsilon, Z_0)$



7. Obtain a PDA for the language $L = \{a^i b^j c^k \mid i+j=k, i \& j \geq 0\}$

$\delta(q_0, a, Z_0) = (q_0, aZ_0)$ } Initialize
$\delta(q_0, \epsilon, Z_0) = (q_f, Z_0)$

$\delta(q_0, a, a) = (q_0, aa)$ } Push
$\delta(q_0, b, b) = (q_0, bb)$
$\delta(q_0, b, a) = (q_0, ba)$

$\delta(q_0, c, b) = (q_1, \epsilon)$ } Pop
$\delta(q_0, c, a) = (q_1, \epsilon)$
$\delta(q_1, c, b) = (q_1, \epsilon)$
$\delta(q_1, c, a) = (q_1, \epsilon)$

$\delta(q_1, \epsilon, Z_0) = (q_f, Z_0)$

ID moves :- aabbcccc

$(q_0, aabbcccc, Z_0) \vdash (q_0, abbcccc, aZ_0)$

$\vdash (q_0, bbcccc, aaZ_0)$

$\vdash (q_0, bcccc, baaZ_0)$

$\vdash (q_0, cccc, bbaaZ_0)$

$\vdash (q_1, ccc, baaZ_0)$

$\vdash (q_1, cc, aaZ_0)$

$\vdash (q_1, c, aZ_0)$

$\vdash (q_f, \epsilon, Z_0)$

String is accepted as the stack is empty

Convertion from grammar to PDA:- [CFG to PDA]

For each variable A

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

Terminal 'a'

$\delta(q, a, a) = \{(q, \epsilon)\}$

1. Convert the given CFG into PDA

i) $E \to E+E \mid E*E \mid (E) \mid I$

$I \to a \mid b \mid Ia \mid Ib \mid I0 \mid I1$

$\delta(q, \epsilon, E) = \{(q, E+E), (q, E*E), (q, (E)), (q, I)\}$

$\delta(q, \epsilon, I) = \{(q, a), (q, b), (q, Ia), (q, Ib), (q, I0), (q, I1)\}$

$\delta(q, +, +) = (q, \epsilon)$

$\delta(q, *, *) = (q, \epsilon)$

$\delta(q, (, () = (q, \epsilon)$

$\delta(q,),)) = (q, \epsilon)$

$\delta(q, a, a) = (q, \epsilon)$

$\delta(q, b, b) = (q, \epsilon)$

$\delta(q, 0, 0) = (q, \epsilon)$

$\delta(q, 1, 1) = (q, \epsilon)$

ii) $S \to aABC$

$A \to aB \mid a$

$B \to bA \mid b$

$C \to a$

$\delta(q, \epsilon, S) = \{(q, aABC)\}$

$\delta(q, \epsilon, A) = \{(q, aB), (q, a)\}$

$\delta(q, \epsilon, B) = \{(q, bA), (q, b)\}$

$\delta(q, \epsilon, C) = (q, a)$

$\delta(q, a, a) = (q, \epsilon)$

$\delta(q, b, b) = (q, \epsilon)$

iii) $S \longrightarrow aABB \mid aAA$

$A \longrightarrow aBB \mid a$

$B \longrightarrow bBB \mid A$

$C \longrightarrow a$

$\delta(q, \epsilon, S) = \{(q, aABB), (q, aAA)\}$

$\delta(q, \epsilon, A) = \{(q, aBB), (q, a)\}$

$\delta(q, \epsilon, B) = \{(q, bBB), (q, A)\}$

$\delta(q, \epsilon, C) = (q, a)$

$\delta(q, a, a) = (q, \epsilon)$

$\delta(q, b, b) = (q, \epsilon)$