Name - Syeda Reeha Quasar   group - 6C7

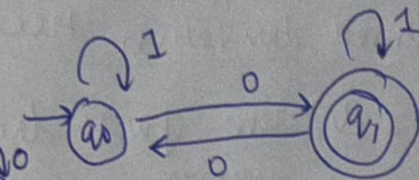
Roll no. - 14114802719

Assignment - 1

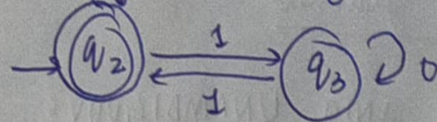
COMPILER DESIGN

Ques 1. Construct finite automata that contains odd no. of 0's and even number of 1's

Ans DFA for odd no. of 0's :-



DFA for even no. of 1's :-

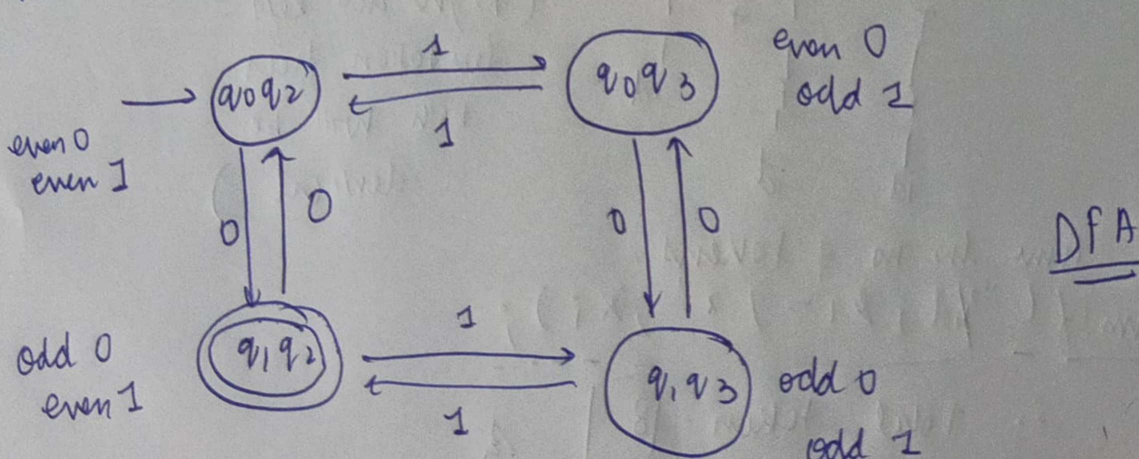


To merge the 2 machines we will take the cartesian product of the states of these 2 machines.

Initial state of these DFA will be state which contains the initial states of those sepracate machines. As q_0 and q_2 are initial states thus q_0q_2 is initial state of DFA :-

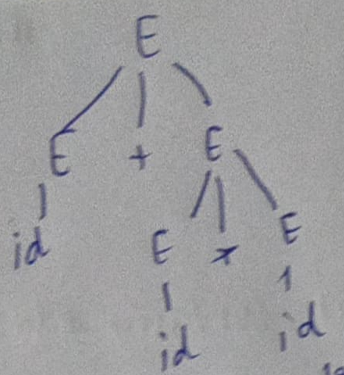
For odd no. of 0's and even no. of 1's :

as q_1 indicates odd no. of 0's and q_2 indicates even no. of 1's. So the final state of each required DFA will contain both q_1 and q_2. Final state = $\{q_1 q_2\}$

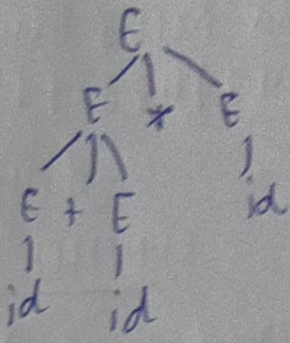


Ques 2. Show that the given grammar is ambiguous for the string "id + id * id". Also, find an equivalent unambiguous grammar. $E \rightarrow E+E \mid E*E \mid id$

Ans To check ambiguity we will start with constructing rightmost and leftmost derivation tree.



rightmost derivation tree



leftmost derivation tree

Since 2 different parse tree for same grammar, so it is ambiguous grammar.

TO CONVERT INTO UNAMBIGUOUS GRAMMAR:

As there are 2 different operators +, * (* has higher precedence) we should take care that highest precedence operator should be at the last level

So we will introduce several different symbols like:

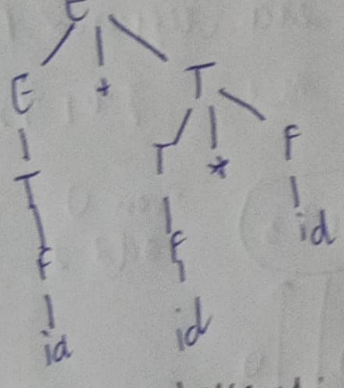
$E \to E+T \mid T$

$T \to T*F \mid F$

$F \to id$

once we reach T, we cannot generate any +

So, we are taking care of precedence by defining different levels.



(Right and left derivation trees both will be identical)

Ans 3. Count the no. of tokens:-

printf ("%d, &i = %x" , x i) ;

∴ Total token = $\boxed{8}$ Ans

" " → counted as 1 token rest all keywords)

characters counted.

Ans 4. Consider the following grammar.

$S \to iEtSeS \mid iEtS \mid a$

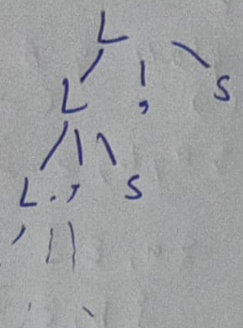
$E \to b$

Check grammar for left recursion or not?

Ans 4. A production grammar is said to be having left recursion if the leftmost variable of its RHS is same as variables of its LHS.

$$S \rightarrow S\alpha \mid \epsilon$$

Since, $L \rightarrow L;S$ it is left recursive grammar.



Removing left recursion : $L \rightarrow SL'$
$L' = ;SL' \mid \epsilon$

Ans 5. Perform the left factoring for the following grammar.

$S \rightarrow iEtSeS \mid iEtS \mid a$
$E \rightarrow b$

Ans In the first two productions for S, iEtS part is common for both the productions so.

1. $S \rightarrow iEtSS' \mid a$
2. $S' \rightarrow eS \mid \epsilon$
3. $E \rightarrow b$

Ans 6. Find first and follow for the following grammar

$S \rightarrow ABCDE$
$A \rightarrow a / \epsilon$
$B \rightarrow b / \epsilon$
$C \rightarrow c$
$D \rightarrow d / \epsilon$
$E \rightarrow e / \epsilon$

Production	first	follow
$S \rightarrow ABCDE$	{a, b, c}	{\$}
$A \rightarrow a/\epsilon$	{a, ε}	{b, c}
$B \rightarrow b/\epsilon$	{b, ε}	{c}
$C \rightarrow c$	{c}	{d, e, \$}
$D \rightarrow d/\epsilon$	{d, ε}	{e, \$}
$E \rightarrow e/\epsilon$	{e, ε}	{\$}

FIRST

① for S

→ First of S : First of {ABCDE}

∵ $A \to a / \varepsilon$

substituting ε in place of A

First (B)

∵ $B \to b / \varepsilon$

substituting ε in place of B

First (C) ∵ $C \to c$

We get first of (S) = {a, b, c}

→ First (A) = $A \to a / \varepsilon$ = {a, ε} (∵ No non-terminal)

→ First (B) = $B \to b / \varepsilon$ = {b, ε}

→ (First (C)) = $C \to c$ = {c}

→ First (D) = $D \to d / \varepsilon$ = {d, ε}

→ First (E) = $E \to e / \varepsilon$ = {e, ε}

FOLLOW

① Follow (S) = {\$}

∵ S is not present in any RHS of productions and it is at start, it has \$ has follow.

② Follow (A) = First (B) (from production $S \to ABCDE$)

= {b, ε}

substituting ε in place of B, we get

first of C = {c}

Follow (A) = {b, c}

③ Follow of (B) = first of (C) = {c}

④ Follow (C) = First (D) = {d, ε}

⇒ {d, e, \$} (follow of S and first of E)

⑤ follow (D) = first (E) = {e, ε} ⇒ {e, \$}

⑥ Follow (E) = Follow (S) ($S \to ABCDE$)

= {\$}