06/10/2022
WEDNESDAY

CS6109 - ASSIGNMENT-II

NAME: ASHWIN MUTHURAMAN A

REGISTER NUMBER: 2020103005

SUBJECT CODE: CS6109

SUBJECT TITLE: COMPILER DESIGN

1) Consider the below given grammar and answer the following questions. (The grammar is already augmented)

(0) S → Stmts
(1) Stmts → Stmt
(2) Stmts → Stmts ; Stmt
(3) Stmt → Var = E
(4) Var → id [E]
(5) Var → id
(6) E → id
(7) E → (E)

(a) Construct the set of LR(0) items and the DFA capable of recognizing it.

Ans: The LR(0) item sets are as follows:

I_0: S → .Stmts
Stmts → .Stmt
Stmts → .Stmts; Stmt
Stmt → .Var = E
Var → .id [E]
Var → .id

I_1: S → Stmts.
Stmts → Stmts. ; Stmt

I_2: Stmts → Stmt.

I_3: Stmt → Var. = E

2020103005

I_4: Var → id.[E]
Var → id.

I_5: Stmts → Stmts ;. Stmt
Stmt → .Var = E
Var → .id[E]
Var → .id

I_6: Stmt → Var =.E
E → .id
E → .(E)

I_7: Var → id[.E]
E → .id
E → .(E)

I_8: Stmts → Stmts ; Stmt.

I_9: Stmt → Var = E.

I_{10}: E → id.

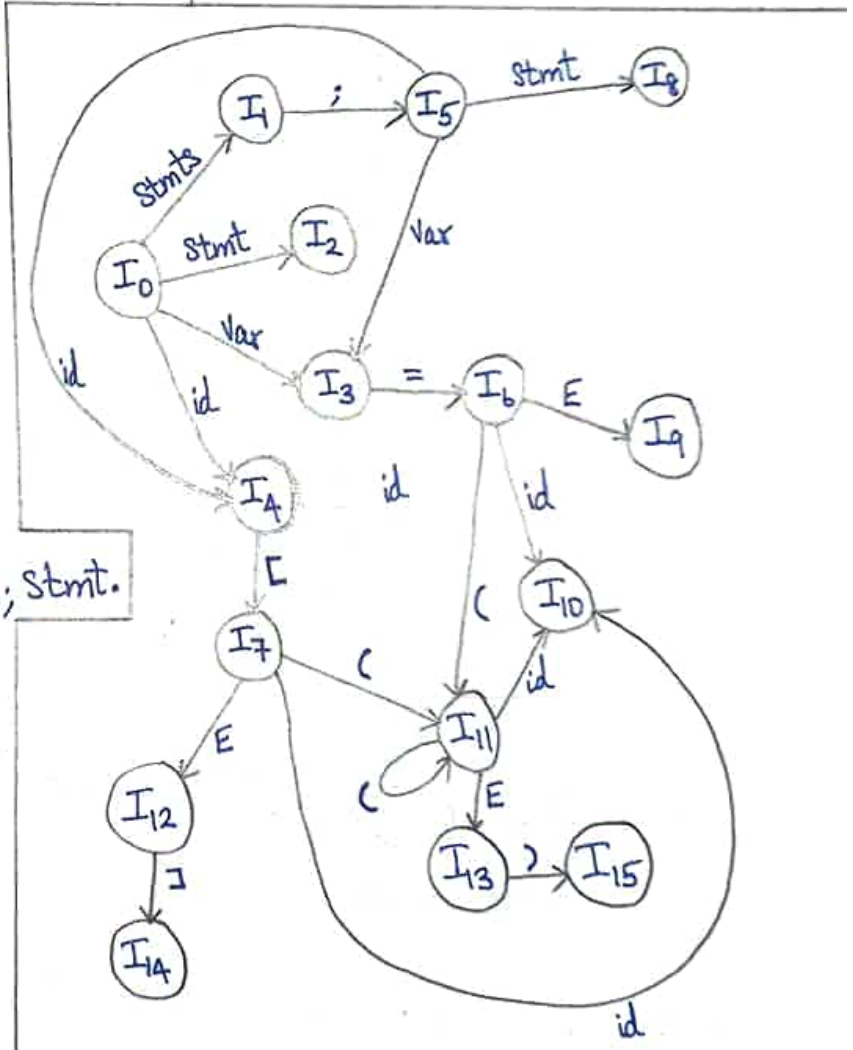
I_{11}: E → (.E)
E → .id
E → .(E)

I_{12}: Var → id[E.]

I_{13}: E → (E.)

I_{14}: Var → id[E].

I_{15}: E → (E).



DFA recognizing LR(0) items for the given grammar.

2020103005

(b) Construct the SLR(0) parsing table and determine if this grammar is LR(0). Justify.

Ans: The SLR(0) parsing table is as follows:

State	Action								Goto			
	;	=	id	[	]	(	)	$	Stmts	Stmt	Var	E
0			S_4						1	2	3	
1	S_5							accept				
2	r_1	r_1	r_1	r_1	r_1	r_1	r_1	r_1				
3		S_6										
4	r_5	r_5	r_5	S_7/r_5	r_5	r_5	r_5	r_5				
5			S_4							8		
6			S_{10}			S_{11}						9
7			S_{10}			S_{11}						12
8	r_2	r_2	r_2	r_2	r_2	r_2	r_2	r_2				
9	r_3	r_3	r_3	r_3	r_3	r_3	r_3	r_3				
10	r_6	r_6	r_6	r_6	r_6	r_6	r_6	r_6				
11			S_{10}			S_{11}						13
12					S_4							
13							S_{15}					
14	r_4	r_4	r_4	r_4	r_4	r_4	r_4	r_4				
15	r_7	r_7	r_7	r_7	r_7	r_7	r_7	r_7				

Since there is a shift-reduce conflict in Action$[I_4,]$, the given grammar is not LR(0).

2020103005

(c) Is the SLR(1) DFA for this grammar the same as the LR(0) DFA? Why?

Ans: Yes, the SLR(1) DFA will be the same as LR(0) DFA. The difference in these methods can be seen only in the parsing table. In SLR(1), the reduce action $(A \to \alpha)$ will be there in the terminals which belong to FOLLOW(A), whereas the reduce action $(A \to \alpha)$ in LR(0) will be there in every terminal belonging to the grammar.

(d) Is this grammar SLR(1)? Justify by constructing its table

Ans: The FIRST and FOLLOW for the terminals and variables of grammar are as follows.

NT	FIRST	FOLLOW
Stmts	{id }	{$, ;}
Stmt	{id }	{$, ;}
Var	{id }	{=}
E	{(, id}	{),], $, ;}

FIRST(;) = {;}
FIRST(() = {(}
FIRST()) = {)}
FIRST(id) = {id}
FIRST([) = {[}
FIRST(]) = {]}
FIRST(=) = {=}

2020103005

The SLR(1) parsing table is as follows:

State	Action ;	=	id	[	]	(	)	$	Goto Stmts	Stmt	Var	E
0			S_4						1	2	3	
1	S_5							acept				
2	r_1							r_1				
3		S_6										
4		r_5		S_7								
5			S_4							8		
6			S_{10}			S_{11}						9
7			S_{10}			S_{11}						12
8	r_2							r_2				
9	r_3							r_3				
10	r_6				r_6		r_6	r_6				
11			S_{10}			S_{11}						13
12					S_{14}							
13							S_{15}					
14		r_4										
15	r_7				r_7		r_7	r_7				

Yes, the grammar is SLR(1), as there is no conflict scenario in SLR(1) parsing table.

(e) Construct the set of LR(1) items and the DFA capable of recognizing it.

Ans: The LR(1) item sets are as follows:

2020103005

I_0: S → .Stmts, $
Stmts → .Stmt, $/;
Stmts → . Stmts; Stmt, $/;
Stmt → .Var = E, $/;
Var → .id[E], =
Var → .id, =

I_1: S → Stmts ., $
Stmts → Stmts .; Stmt, $/;

I_2: Stmts → Stmt ., $/;

I_3: Stmt → Var .= E, $/;

I_4: Var → id .[E], =
Var → id ., =

I_5: Stmts → Stmts; . Stmt, $/;
Stmt → .Var = E, $/;
Var → .id[E], =
Var → .id, =

I_6: Stmt → Var = . E, $/;
E → .id, $/;
E → .(E), $/;

I_7: Var → id[.E], =
E → .id,]
E → .(E),]

I_8: Stmts → Stmts; Stmt ., $/;

I_9: Stmt → Var = E ., $/;

I_{10}: E → id ., $/;

I_{11}: E → (.E), $/;
E → .id,)
E → .(E),)

I_{12}: Var → id[E .], =

I_{13}: E → id .,]

I_{14}: E → (.E),]
E → .id,)
E → .(E),)

I_{15}: E → (E .), $/;

I_{16}: E → id .,)

I_{17}: E → (.E),)
E → .id,)
E → .(E),)

I_{18}: Var → id[E] ., =

I_{19}: E → (E .),]

I_{20}: E → (E) ., $/;

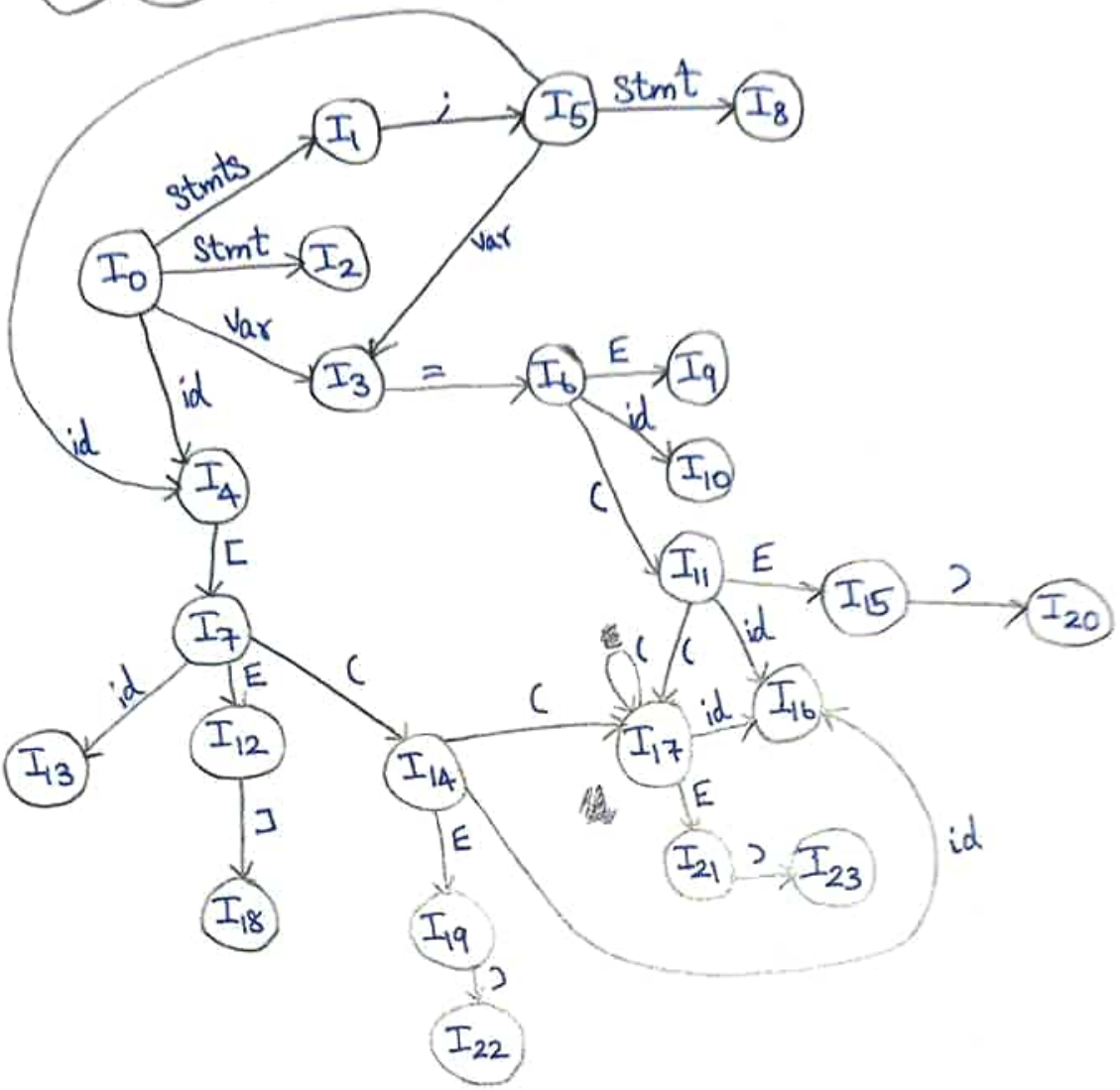
I_{21}: E → (E .),)

I_{22}: E → (E) .,]

I_{23}: E → (E) .,)

2020103005

LR(1) DFA is as follows:



(f) Construct the LR(1) parsing table and determine if this grammar is LR(1). Justify.

Ans: The LR(1) parsing table is as follows:

State	Action								Goto			
	;	=	id	[	]	(	)	$	Stmts	Stmt	Var	E
0			S_4						1	2	3	
1	S_5							accept				
2	r_1							r_1				
3		S_6										

2020103005

State	Action ;	=	id	[	]	(	)	$	Goto Stmts	Stmt	Var	E
4		r_5		S_7								
5			S_4							8		
6			S_{10}			S_{11}						9
7			S_{13}			S_{14}						12
8	r_2							r_2				
9	r_3							r_3				
10	r_6							r_6				
11			S_{16}			S_{17}						15
12					S_{18}							
13					r_6							
14			S_{16}			S_{17}						19
15							S_{20}					
16							r_6					
17			S_{16}			S_{17}						21
18		r_4										
19							S_{22}					
20	r_7							r_7				
21							S_{23}					
22					r_7							
23							r_7					

Since, there is no conflicting scenario in LR(1) parsing table, the given grammar is LR(1).

(g) How would you derive the LALR(1) parsing table ~~for~~ this grammar? What is the difference between this and the table found above?

2020103005

To generate LALR(1) parsing table, we combine the common cores to single state.

$J_0 = I_0$, $J_4 = I_4$, $J_8 = I_8$, $J_{12} = I_{12}$,

$J_1 = I_1$, $J_5 = I_5$, $J_9 = I_9$, $J_{13} = I_{15} I_{19} I_{21}$,

$J_2 = I_2$, $J_6 = I_6$, $J_{10} = I_{10} I_{13} I_{16}$, $J_{14} = I_{18}$

$J_3 = I_3$, $J_7 = I_7$, $J_{11} = I_{11} I_{14} I_{17}$, $J_{15} = I_{20} I_{22} I_{23}$

The LALR(1) parsing table w.r.t. $I_0 \dots I_{23}$ is as follows:

	Action								Goto			
State	;	=	id	[	]	(	)	$	Stmts	Stmt	Var	E
0			S_4						1	2	3	
1	S_5							accept				
2	r_1							r_1				
3		S_6										
4		r_5		S_7								
5			S_4							8		
6			$S_{10,13,16}$			$S_{11,14,17}$						9
7			$S_{10,13,16}$			$S_{11,14,17}$						12
8	r_2							r_2				
9	r_3							r_3				
10,13,16	r_6				r_6		r_6	r_6				
11,14,17			$S_{10,13,16}$			$S_{11,14,17}$						15,19,21
12					S_{18}							
15,19,21							$S_{20,22,23}$					
18		r_4										
20,22,23	r_7				r_7		r_7	r_7				

2020103005

Since, there is no conflict in LALR(1) parsing table, the given grammar is LALR(1). The difference between this table and the previous one is that the number of states in LALR(1) is lesser than LR(1).

2) Consider the below given grammar and answer the following questions: (The grammar is already augmented).

Ans

(0) $S' \rightarrow S$
(1) $S \rightarrow id = E;$
(2) $S \rightarrow L = E;$
(3) $E \rightarrow E + E$
(4) $E \rightarrow id$
(5) $E \rightarrow L$
(6) $L \rightarrow id[E]$
(7) $L \rightarrow [E]$

(a) Construct the set of LR(0) items and the DFA capable of recognizing it.

Ans. The LR(0) item sets are as follows:

I_0: $S' \rightarrow .S$
$S \rightarrow .id = E;$
$S \rightarrow .L = E;$
$L \rightarrow .id[E]$
$L \rightarrow .[E]$

I_1: $S' \rightarrow S.$

I_2: $S \rightarrow id.=E;$
$L \rightarrow id.[E]$

I_3: $S \rightarrow L.=E;$
$L \rightarrow [.E]$

I_4: $E \rightarrow .E+E$
$E \rightarrow .id$
$E \rightarrow .L$
$L \rightarrow .id[E]$
$L \rightarrow .[E]$

I_5: $S \rightarrow id =.E;$
$E \rightarrow .id$
$E \rightarrow .L$
$E \rightarrow .E+E$
$L \rightarrow .id[E]$
$L \rightarrow .[E]$

2020103005

I_6: L → id [E]
E → .E+E
E → .id
E → .L
L → .id[E]
L → .[E]

I_7: S → L=.E;
E → .E+E
E → .id
E → .L
L → .id[E]
L → .[E]

I_8: L → [E.]
E → E.+E

I_9: E → id.
L → id.[E]

I_{10}: E → L.

I_{11}:
S → id=E.;
E → E.+E

I_{12}: L → id [E.]
E → E.+E

I_{13}: S → L=E.;
E → E.+E

I_{14}: L → [E].

I_{15}: E → E+.E
E → .E+E
E → .id
E → .L
L → .id[E]
L → .[E]

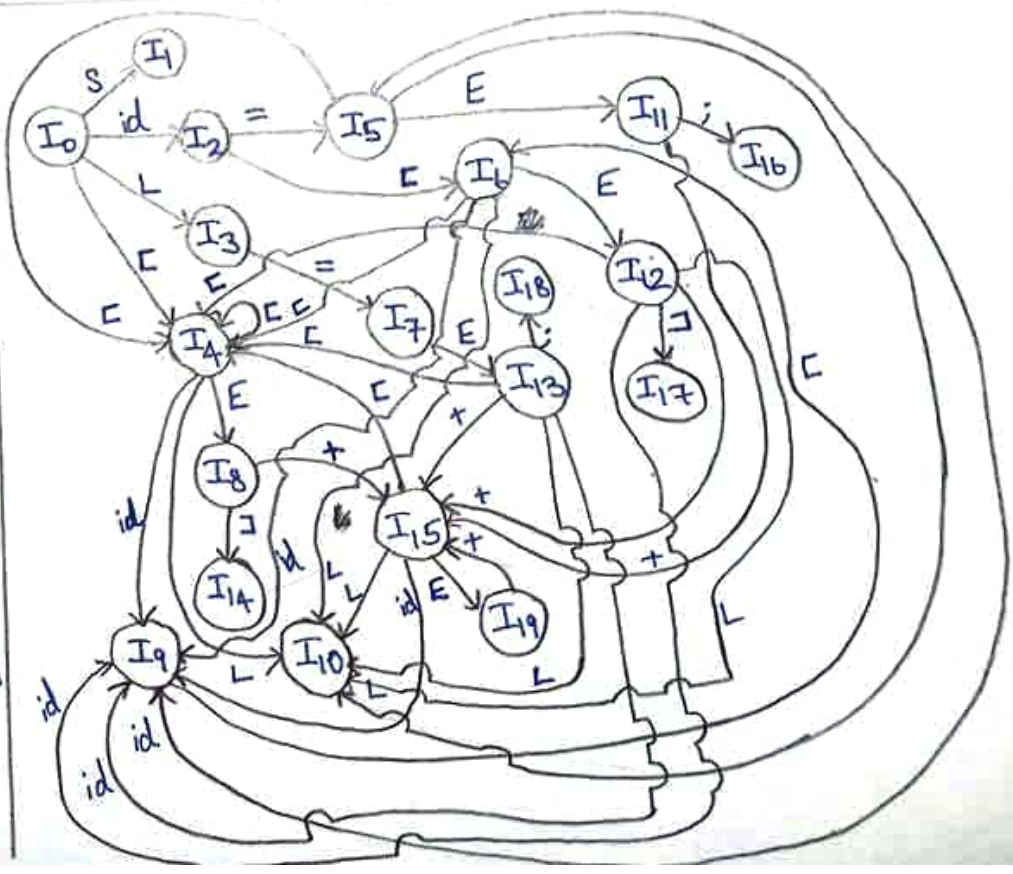
I_{16}: S → id=E;.

I_{17}: L → id [E].

I_{18}: S → L=E;.

I_{19}: E → E+E.
E → E.+E

LR(0) DFA:



2020103005

(b) Construct the SLR(0) parsing table and determine if this grammar is LR(0). Justify.

Ans: The SLR(0) parsing table is as follows:

	Action								Goto	
State	id	=	;	+	[	]	$	S	E	L
0	S_2				S_4			1		3
1							accept			
2		S_5			S_6					
3		S_7								
4	S_9				S_4				8	10
5	S_9				S_4				11	10
6	S_9				S_4				12	10
7	S_9				S_4				13	10
8				S_{15}		S_{14}				
9	r_4	r_4	r_4	r_4	S_6/r_4	r_4	r_4			
10	r_5	r_5	r_5	r_5	r_5	r_5	r_5			
11			S_{16}	S_{15}						
12				S_{15}		S_{17}				
13			S_{18}	S_{15}						
14	r_7	r_7	r_7	r_7	r_7	r_7	r_7			
15	S_9				S_4				19	10
16	r_1	r_1	r_1	r_1	r_1	r_1	r_1			
17	r_6	r_6	r_6	r_6	r_6	r_6	r_6			
18	r_2	r_2	r_2	r_2	r_2	r_2	r_2			
19	r_3	r_3	r_3	S_{15}/r_3	r_3	r_3	r_3			

As, there are 2 shift-reduce conflicts in the table, we conclude that the given grammar is not LR(0).

2020103005

(c) If the grammar is not SLR(0), then is it LR(1)? Justify by constructing LR(1) table only if it is not SLR(0).

Ans: The LR(1) item sets are as follows:

I_0: $S' \to .S, \$$
$S \to .id=E;, \$$
$S \to .L=E;, \$$
$L \to .id[E], =$
$L \to .[E], =$

I_1: $S' \to S., \$$

I_2: $S \to id.=E;, \$$
$L \to id.[E], =$

I_3: $S \to L.=E;, \$$

I_4: $L \to [.E], =$
$E \to .E+E,]/+$
$E \to .id,]/+$
$E \to .L,]/+$
$L \to .id[E],]/+$
$L \to .[E],]/+$

I_5: $S \to id=.E;, \$$
$E \to .E+E, ;/+$
$E \to .id, ;/+$
$E \to .L, ;/+$
$L \to .id[E], ;/+$
$L \to .[E], ;/+$

I_6: $L \to id[.E], =$
$E \to .E+E,]/+$
$E \to .id,]/+$
$E \to .L,]/+$
$L \to .id[E],]/+$
$L \to .[E],]/+$

I_7: $S \to L=.E;, \$$
$E \to .E+E, ;/+$
$E \to .id, ;/+$
$E \to .L, ;/+$
$L \to .id[E], ;/+$
$L \to .[E], ;/+$

I_8: $L \to [E.], =$
$E \to E.+E,]/+$

I_9: $E \to id.,]/+$
$L \to id.[E],]/+$

I_{10}: $E \to L.,]/+$

I_{11}: $L \to [.E],]/+$
$E \to .E+E,]/+$
$E \to .id,]/+$
$E \to .L,]/+$
$L \to .id[E],]/+$
$L \to .[E],]/+$

I_{12}: $S \to id=E.;, \$$
$E \to E.+E, ;/+$

I_{13}: $E \to id., ;/+$
$L \to id.[E], ;/+$

I_{14}: $E \to L., ;/+$

I_{15}: $L \to [.E], ;/+$
$E \to .E+E,]/+$
$E \to .id,]/+$
$E \to .L,]/+$
$L \to .id[E],]/+$
$L \to .[E],]/+$

I_{16}: $L \to id[E.], =$
$E \to E.+E,]/+$

I_{17}: $S \to L=E.;, \$$
$E \to E.+E, ;/+$

I_{18}: $L \to [E]., =$

I_{19}: $E \to E+.E,]/+$
$E \to .E+E,]/+$
$E \to .id,]/+$
$E \to .L,]/+$
$L \to .id[E],]/+$
$L \to .[E],]/+$

I_{20}: $L \to id[\cdot E],]/+$
$E \to \cdot E+E,]/+$
$E \to \cdot id,]/+$
$E \to \cdot L,]/+$
$L \to \cdot id[E],]/+$
$L \to \cdot [E],]/+$

I_{21}: $L \to [E\cdot],]/+$
$E \to E\cdot +E,]/+$

I_{22}: $S \to id = E;\cdot, \$$

I_{23}: $E \to E+\cdot E, ;/+$
$E \to \cdot E+E, ;/+$
$E \to \cdot id, ;/+$
$E \to \cdot L, ;/+$
$L \to \cdot id[E], ;/+$
$L \to \cdot [E], ;/+$

I_{24}: $L \to id[\cdot E], ;/+$
$E \to \cdot E+E,]/+$
$E \to \cdot id,]/+$
$E \to \cdot L,]/+$
$L \to \cdot id[E],]/+$
$L \to \cdot [E],]/+$

I_{25}: $L \to [E\cdot], ;/+$
$E \to E\cdot +E,]/+$

I_{26}: $L \to id[E]\cdot, =$

I_{27}: $S \to L=E;\cdot, \$$

I_{28}: $E \to E+E\cdot,]/+$
$E \to E\cdot +E,]/+$

I_{29}: $L \to id[E\cdot],]/+$
$E \to E\cdot +E,]/+$

I_{30}: $L \to [E]\cdot,]/+$

I_{31}: $E \to E+E\cdot, ;/+$
$E \to E\cdot +E, ;/+$

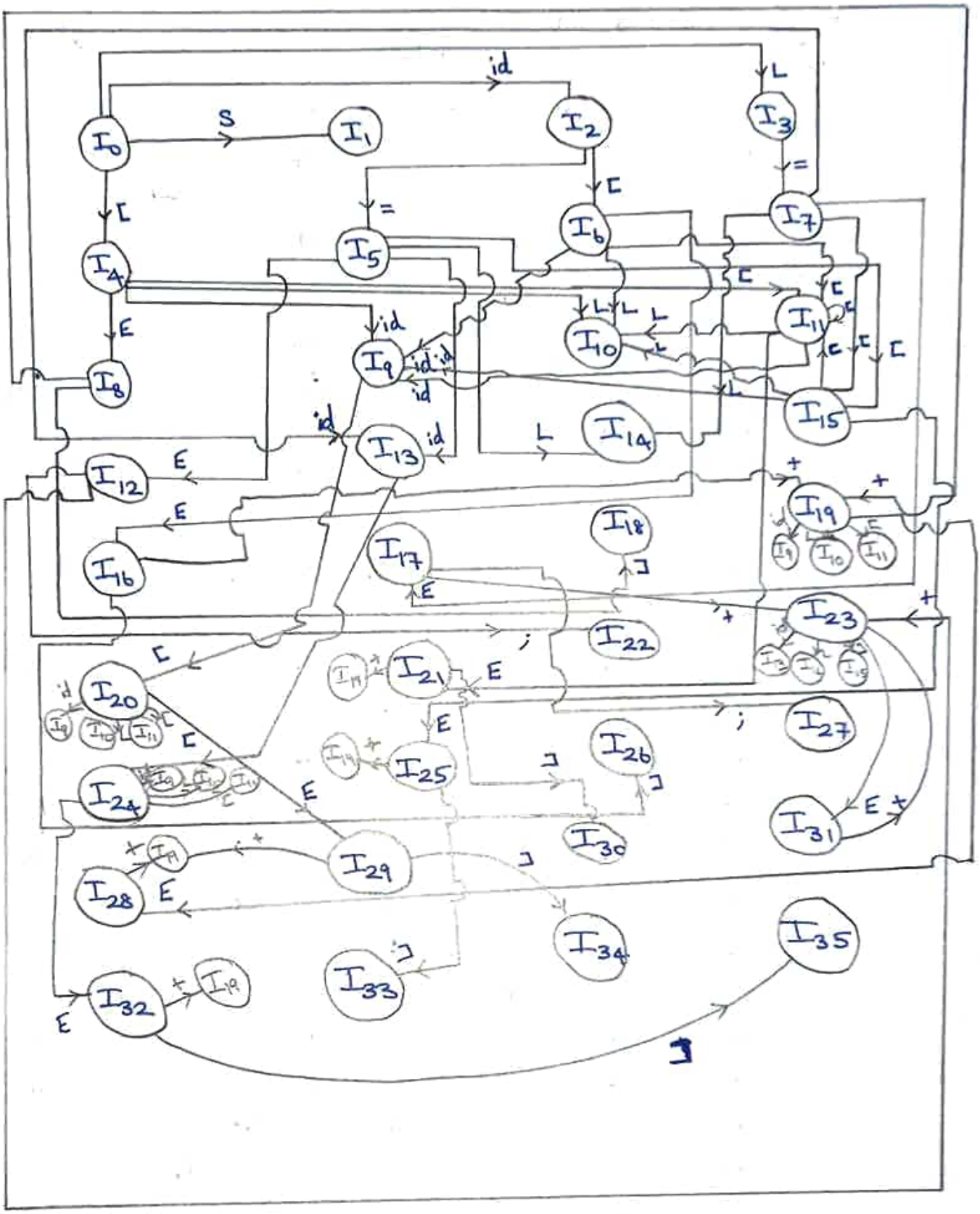
I_{32}: $L \to id[E\cdot], ;/+$
$E \to E\cdot +E,]/+$

I_{33}: $L \to [E]\cdot, ;/+$

I_{34}: $L \to id[E]\cdot,]/+$

I_{35}: $L \to id[E]\cdot, ;/+$

2020103005



LR(1) DFA for given grammar.

2020103005

LR(1) parsing table:

State	Action							Goto		
	id	=	;	+	[	]	$	S	E	L
0	S_2				S_4			1		3
1							accept			
2		S_5			S_6					
3		S_7								
4	S_9				S_{11}				8	10
5	S_{13}				S_{15}				12	14
6	S_9				S_{11}				16	10
7	S_{13}				S_{15}				17	14
8				S_{19}		S_{18}				
9				r_4	S_{20}	r_4				
10				r_5		r_5				
11	S_9				S_{11}				21	10
12			S_{22}	S_{23}						
13			r_4	r_4	S_{24}					
14			r_5	r_5						
15	S_9				S_{11}				25	10
16				S_{19}		S_{26}				
17			S_{27}	S_{23}						
18		r_7								
19	S_9				S_{11}				28	10
20	S_9				S_{11}				29	10
21				S_{19}		S_{30}				
22							r_1			
23	S_{13}				S_{15}				31	14
24	S_9				S_{11}				32	10
25				S_{19}		S_{33}				
26		r_6								
27							r_2			

2020103005

State	Action							Goto		
	id	=	;	+	[	]	$	S	E	L
28				S_{19}/r_3		r_3				
29				S_{19}		S_{34}				
30				r_7		r_7				
31			r_3	S_{28}/r_3						
32				S_{19}		S_{35}				
33			r_7	r_7						
34				r_6		r_6				
35			r_6	r_6						

Since there are 2 shift-reduce conflicts in LR(1) parsing table, the given grammar is not LR(1).

(d) Parse the input string "a = b[q][v] + c;" for given grammar and show the stack trace.

Ans: Given input: a = b[q][v] + c;

⇒ Input string: id = id [id] [id] + id;

Line	Stack	Input	Action
(1)	0	id=id[id][id]+id;$	S_2
(2)	02	=id[id][id]+id;$	S_5
(3)	025	id[id][id]+id;$	S_{13}
(4)	025 13	[id][id]+id;$	S_{24}
(5)	025 13 24	id][id]+id;$	S_9

2020103005

Line	Stack	Input	Action
(6)	0 2 5 13 24 9	][id]+id;$	r_4
(7)	0 2 5 13 24	][id]+id;$	S_{32}
(8)	0 2 5 13 24	][id]+id;$	S_{35}
(9)	0 2 5 13 24 32	[id]+id;$	—
(10)	0 2 5 13 24 32 35	id]+id;$	

Action $[I_{35}, id]$ = error.

∴ Given input string does not belong to the grammar //