Wednesday
13/11/24

$E' \rightarrow E$

① $E \rightarrow E+T$

② $E \rightarrow T$

③ $T \rightarrow T*F$

④ $T \rightarrow F$

⑤ $F \rightarrow (E)$

⑥ $F \rightarrow id$

Step 4: Construct Parsing Table

Action table says → what to do if we get this token (terminal symbol)
- whether to shift or reduce
- whether to accept or error

GoTo table says → where to go after a reduction

Rows → for each of the states.

Columns → all the tokens including $

State	Action Table						GoTo Table		
	id	+	*	(	)	$	E	T	F
I_0	S5			S4			1	2	3
I_1		S6				accept			
I_2		R2	S7		R2	R2			
I_3		R4	R4		R4	R4			
I_4	S5			S4			8	2	3
I_5		R6	R6		R6	R6			
I_6	S5			S4				9	3
I_7	S5			S4					10
I_8		S6			S11				
I_9		R1	S7		R1	R1			
I_{10}		R3	R3		R3	R3			
I_{11}		R5	R5		R5	R5			

* It is not possible to see E' since not in body ∴ not in GoTo table.

* whenever we have a transition from one state to another on a terminal symbol, shift the destination state in stack.

S 5
↙ ↘
shift state no

* To reduce we have to check if there is any item with the cursor at the end.

* If I have an production rule. with cursor at end → apply reduction for all the terminal symbols of the head of the production rule.

* For $E' \to E$ we have only $ in followset of E', so we consider it as special case and do not reduce.

* For I_2, we have $E \to T.$ ∴ E's follows are

$\$, +,)$ we will reduce for all of them.

R G
↙ ↳ no of production rule
Reduce

② $E \to T$
③ $T \to F$

* Blank spaces indicate errors, I_3 → কোন ধরণের error.

S7, R2 ← Shift Reduce conflict
parser doesnot know whether to shift or reduce. just by analyzing the grammar.
we need complex parsers for that.

There can also be reduce - reduce conflict.

GoTo table

To কোন non-terminal এর কোন state → যাবে.

→ total tokens (terminals)

* No of colums to SLR(1) → $6P_1$

" " " " " (2) → $6P_2$

{ The automaton is LR(0)
{ The parsen table is SLR(1)