Asian Institute of Technology

Calculator Project

An assignment submitted in partial fulfillment of the
requirements for AT70.07
Programming Language and Compilers

Submitted To:
Professor: Phan Minh Dung

Submitted By:
Sunil Prajapati (st124073)

31st March 2024

Parser:

Chosen Parser: Bottom-up SLR(1) parser

Grammar

The grammar rules for the calculator are defined below:

R0: S' → S

R1: S → S * T

R2: S → T

R3: T → T + F

R4: T → F

R5: F → N

Hence the canonical LR(0) items are:

State 0

S' →. S

S → . S * T

S → . T

T → . T + F

T → . F

F → . N

State 1

S' → S..

S → S . * T

State 2

S → T .

T → T . + F

State 3

T → F .

State 4

F → N .

State 5

S → S * . T

T → . T + F

T → . F

F → . N

State 6

T → T + . F

F → . N

State 7

E → E * T .

T → T . + F

State 8

T → T + F .

Parsing Table

The parsing table is constructed as below:

	Action				Go To		
State	+	*	N	$	S	T	F
0			s4		1	2	3
1		s5		acc			
2	s6	r2		r2			
3	r4	r4		r4			
4	r5	r5		r5			
5			s4			7	3
6			s4				8
7	s6	r1		r1			
8	r3	r3		r3			

Rules of Translation

1. Value Calculation

Production	Semantic Rules
S → S1 * T	S.val := S1.val * T.val
S → T	S.val := T.val
T → T1 + F	T.val := T1.val + F.val
T → F	T.val := F.val
F → N	F.val := N.lexval

2. Prefix Notation

Production	Semantic Rules
S → S1 * T	S.pf := ‘*’ || S1.pf || T.pf
S → T	S.pf := T.pf
T → T1 + F	T.pf := ‘+’ || T1.pf || F.pf
T → F	T.pf := F.pf
F → N	F.pf := N.pf

3. Postfix Notation

Production	Semantic Rules
S → S * T	S.pf := S.pf || T.pf || ‘*’
S → T	S.pf := T.pf
T → T + F	T.pf := T1.pf || F.pf || ‘+’
T → F	T.pf := F.pf
F → N	F.pf := N.pf

Conclusion

In short, our calculator project uses a special technique called "Bottom-up LR(1) parsing" along with some smart actions to understand and translate input expressions. It can convert expressions into numbers, as well as into simpler forms called "prefix notation" and "postfix notation." This helps us parse expressions accurately and do math calculations effectively.