Chapter 2

Predict Sets

Objective

Today's primary objective:

- To practice creating predict sets
- To create a parsing table from the predict sets

Predict Sets

Predict Sets are needed by top-down (i.e. recursive descent) parsers

- Subroutine version
- Table driven

Given a non-terminal symbol and a terminal (token), need to predict what production to use

Grammar

1. program → stmt_list $$
2. stmt_list → stmt stmt_list
3. stmt_list → ε
4. stmt → id := expr
5. stmt → read id
6. stmt → write expr
7. expr → term term_tail
8. term_tail → add_op term term_tail
9. term_tail → ε
10. term → factor factor_tail
11. factor_tail → mult_op factor factor_tail
12. factor_tail → ε
13. factor → (expr)
14. factor → id
15. factor → number
16. add_op → +
17. add_op → -
18. mult_op → *
19. mult_op → /

Predict Sets

	EPS	FIRST	FOLLOW
program			
stmt_list			
stmt			
expr			
term_tail			
term			
factor_tail			
factor			
add_op			
mult_op			
$$			
id			
:=			
read			
write			
(			
)			
number			
+			
-			
*			
/			

Predict Sets

	EPS	FIRST	FOLLOW
program	false	{$$, id, read, write}	{}
stmt_list	true	{ id, read, write }	{$$}
stmt	false	{id, read, write}	{ id, read, write, $ $}
expr	false	{(, id, number }	{), id, read, write, $$ }
term_tail	true	{+,-}	{), id, read, write, $$ }
term	false	{(, id, number }	{+, -,), id, read, write, $ $}
factor_tail	true	{*, /}	{+, -,), id, read, write, $$ }
factor	false	{(, id, number }	{*, /, +, -,), id, read, write, $$ }
add_op	false	{+, - }	{(, id, number }
mult_op	false	{*, / }	{(, id, number }
$$	false	{$$ }	{}
id	false	{ id }	{+, -, *, /,), :=, id, read, write, $$ }
:=	false	{:= }	{(, id, number }
read	false	{ read }	{id}
write	false	{ write }	{(, id, number }
(	false	{(}	{(, id, number }
)	false	{) }	{+, -, *, /,), id, read, write, $$ }
number	false	{ number }	{+, -, *, /,), id, read, write, $$ }

FIRST/FOLLOWS Sets

1. program → stmt_list $ {id, read, write }
2. stmt_list → stmt stmt_list {id, read, write}
3. stmt_list → ε {$$}
4. stmt → id := expr {id}
5. stmt → read id {read}
6. stmt → write expr {write}
7. expr → term term_tail {(, id, number }
8. term_tail → add_op term term_tail {+, - }
9. term_tail → ε {), id, read, write , $$ }
10. term → factor factor_tail {(, id, number }
11. factor_tail → mult_op factor factor_tail {*, / }
12. factor_tail → ε {+, -,), id, read, write , $$ }
13. factor → (expr) {(}
14. factor → id {id}
15. factor → number {number}
16. add_op → + {+}
17. add_op → - {-}
18. mult_op → * {*}
19. mult_op → / {/}

Parsing Table for Previous

	id	:=	read	write	(	)	number	+	-	*	/	$$
program	1		1	1								
stmt_list	2		2	2								3
stmt	4		5	6								
expr	7				7		7					
term_tail	9		9	9		9		8	8			9
term	10				10		10					
factor_tail	12		12	12				12	12	11	11	12
factor	14				13		15					
add_op								16	17			
mult_op										18	19	