

Context-free Grammar Expression

BNF and BNF w/o Left Recursion for LL(1) Parser

Compiler Theory - Programming Project 3 : Recursive Descent Parser

21400646 Lim Chae Eon

BNF Expression

Italic word : Non-Terminal, **blue colored word** : Terminal, **Yellow box** : left recursion occurred or left factored

1	# Script Definition part		
2	<i>Program</i>	→	<script_start> <i>stmt-sequence</i> <script_end>
3	<i>stmt-sequence</i>	→	<i>statement</i> ; <i>stmt-sequence</i> comment nextLine ε
4	<i>statement</i>	→	<i>if-stmt</i> <i>declaration-stmt</i> <i>loop-stmt</i> assign-stmt <i>function-stmt</i> <i>switch-stmt</i> increment-stmt break
5	<i>simple-stmt</i>	→	<i>stmt-sequence</i> { <i>stmt-sequence</i> }
6			
7	# If-Conditional Statement Part		
8	<i>if-stmt</i>	→	if <i>function-parameter</i> <i>simple-stmt</i> if <i>function-parameter</i> <i>simple-stmt</i> else <i>simple-stmt</i>
9			
10	# Variable Declaration Statement Part		
11	<i>declaration-stmt</i>	→	var <i>id</i>
12	<i>id</i>	→	<i>assign-stmt</i> <i>id</i> , <i>id</i> IDENTIFIER
13			
14	# Assignment Definition Statement Part		
15	<i>assign-stmt</i>	→	IDENTIFIER <i>assign-op</i> <i>exp</i>
16	<i>assign-op</i>	→	= -- +=
17			

18 # Operation Expression Part

19 *exp* → *simple-exp logic-op simple-exp* | *simple-exp*

20 *logic-op* → < | > | <= | >= | == | !=

21 *simple-exp* → *simple-exp add-op term* | *term*

22 *add-op* → + | −

23 *term* → *term mul-op factor* | *factor*

24 *mul-op* → * | /

25 *factor* → (*exp*) | NUMBER | IDENTIFIER

26

27 # Loop Statement Part

28 *loop-stmt* → for *function-parameter simple-stmt*
| while *function-parameter simple-stmt*

29

30 # Function Statement Part

31 *function-stmt* → *function-keyword function-parameter*

32 *function-keyword* → window.prompt | window | parseFloat
| document.writeln | document.write | document

33

34 # Increment/Decrement Operation Part

35 *increment-stmt* → *increment-op IDENTIFIER* | *IDENTIFIER increment-op*

36 *increment-op* → ++ | --

37

38 # Switch-Conditional Statement Part

39 *switch-stmt* → switch *function-parameter { case-part default-block }*

40 *case-part* → *case-block case-part* | ε

41 *case-block* → *case-condition : stmt-sequence*

42 *case-condition* → **case** *case-parameter*
43 *case-parameter* → **NUMBER** | **LITERAL**
44 *default-block* → **default** : *stmt-sequence* | ϵ

45

46 # Function Parameter Handling Part

47 *function-parameter* → (*exp*) | (*exp* ; *exp* ; *exp*) | (**LITERAL**)

48

49 # Comment Part

50 *comment* → **// anything EOL***

*EOL : End of Line

BNF Expression without Left-Recursion

Italic word : Non-Terminal, **blue colored word** : Terminal, Green box : Resolved left recursion or factored

1	# Script Definition part		
2	<i>Program</i>	→	<script_start> <i>stmt-sequence</i> <script_end>
3	<i>stmt-sequence</i>	→	<i>statement</i> ; <i>stmt-sequence</i> <i>comment</i> nextLine ϵ
4	<i>statement</i>	→	<i>if-stmt</i> <i>declaration-stmt</i> <i>loop-stmt</i> <i>assign-stmt</i> <i>function-stmt</i> <i>switch-stmt</i> <i>increment-stmt</i> break
5	<i>simple-stmt</i>	→	<i>stmt-sequence</i> { <i>stmt-sequence</i> }
6			
7	# If-Conditional Statement Part		
8	<i>if-stmt</i>	→	if <i>function-parameter</i> <i>simple-stmt</i> <i>else-part</i>
9	<i>else-part</i>	→	else <i>simple-stmt</i> ϵ
10			
11	# Variable Declaration Statement Part		
12	<i>declaration-stmt</i>	→	var <i>id</i>
13	<i>id</i>	→	<i>assign-stmt</i> id_2 IDENTIFIER id_2
14	id_2	→	ϵ , <i>id</i> id_2
15			
16	# Assignment Definition Statement Part		
17	<i>assign-stmt</i>	→	IDENTIFIER <i>assign-op</i> <i>exp</i>
18	<i>assign-op</i>	→	= -= +=
19			
20	# Operation Expression Part		
21	<i>exp</i>	→	<i>simple-exp</i> <i>logic-exp</i>

22	<i>logic-exp</i>	→	<i>logic-op simple-exp ε</i>
23	<i>logic-op</i>	→	< > <= >= == !=
24	<i>simple-exp</i>	→	<i>term simple-exp₂</i>
25	<i>simple-exp₂</i>	→	<i>add-op term simple-exp₂ ε</i>
26	<i>add-op</i>	→	+ -
27	<i>term</i>	→	<i>factor term₂</i>
28	<i>term₂</i>	→	<i>mul-op factor term₂ ε</i>
29	<i>mul-op</i>	→	* /
30	<i>factor</i>	→	(exp) NUMBER IDENTIFIER
31			
32	# Loop Statement Part		
33	<i>loop-stmt</i>	→	for <i>function-parameter simple-stmt</i> while <i>function-parameter simple-stmt</i>
34			
35	# Function Statement Part		
36	<i>function-stmt</i>	→	<i>function-keyword function-parameter</i>
37	<i>function-keyword</i>	→	window.prompt window parseFloat document.writeln document.write document
38			
39	# Increment/Decrement Operation Part		
40	<i>increment-stmt</i>	→	<i>increment-op IDENTIFIER IDENTIFIER increment-op</i>
41	<i>increment-op</i>	→	++ --
42			
43	# Switch-Conditional Statement Part		
44	<i>switch-stmt</i>	→	switch <i>function-parameter { case-part default-block }</i>
45	<i>case-part</i>	→	<i>case-block case-part case-block</i>

46 *case-block* → *case-condition* : *stmt-sequence* | ϵ

47 *case-condition* → **case** *case-parameter*

48 *case-parameter* → **NUMBER** | **LITERAL**

49 *default-block* → **default** : *stmt-sequence* | ϵ

50

51 # Function Parameter Handling Part

52 *function-parameter* → (*parameter-part*)

53 *parameter-part* → *exp* *condition-part* | **LITERAL**

54 *condition-part* → ; *exp* *condition-part* | ϵ

55

56 # Comment Part

57 *comment* → **// anything nextLine**