BNF

1. program → declaration-list
2. declaration-list → declaration-list declaration | declaration
3. declaration → var-declaration | fun-declaration
4. var-declaration → type-specifier ID ; | type-specifier ID [NUM] ;
5. type-specifier → int | void
6. fun-declaration → type-specifier ID (params) compound-stmt
7. params → param-list | void
8. param-list → param-list , param | param
9. param → type-specifier ID | type-specifier ID [ ]
10. compound-stmt → { local-declarations statement-list }
11. local-declarations → local-declarations var-declaration | empty
12. statement-list → statement-list statement | empty
13. statement → expression-stmt | compound-stmt | selection-stmt | iteration-stmt | return-stmt
14. expression-stmt → expression ; | ;
15. selection-stmt → if (expression) statement | if (expression) statement else statement
16. iteration-stmt → while (expression) statement
17. return-stmt → return; | return expression;
18. expression → var = expression | simple-expression
19. var → ID | ID [expression]
20. simple-expression — additive-expression relop additive-expression | additive-expression
21. relop → <= | < | > | >= | == | !=
22. additive-expression → additive-expression addop term | term
23. addop → + | -
24. term → term mulop factor | factor
25. mulop → * | /
26. factor → (expression) | var | call | NUM
27. call → ID (args)
28. args → arg-list | empty
29. arg-list → arg-list , expression | expression

EBNF

1. program → declaration-list
2. declaration-list → declaration { declaration }
3. declaration → type-specifier ID **[** ; | [NUM]; | (params) compound-stmt **]**
4. type-specifier → int | void
5. params → param { , param } | void
6. param → type-specifier ID **[** [ ] **]**
7. compound-stmt → { local-declarations statement-list }
8. local-declarations → { declaration }
9. statement-list → { expression ; | compound-stmt | selection-stmt | iteration-stmt | return-stmt }
10. selection-stmt → if (expression) statement-list **[** else statement-list **]**
11. iteration-stmt → while (expression) statement-list
12. return-stmt → return **[** additive-expression **]** ;
13. expression → ID **[** [expression] **]** = expression | additive-expression **[** relop additive-expression **]**
14. relop → <= | < | > | >= | == | !=
15. additive-expression → term { addop term }
16. addop → + | -
17. term → factor { mulop factor }
18. mulop → * | /
19. factor → (expression) | ID **[** [expression] **]** | ID (args) | NUM
20. args → **[** expression **{** , expression **}]**