Compile Project 2024

김도영(20206098), 김현욱

1. CFG G
   1. 문법 수정 사항

수정 전 :   
01:  CODE → VDECL CODE | FDECL CODE | ε

02:  VDECL → vtype id semi | vtype ASSIGN semi

03:  ASSIGN → id assign RHS

04:  RHS → EXPR | literal | character | boolstr

05:  EXPR → EXPR addsub EXPR | EXPR multdiv EXPR

06:  EXPR → lparen EXPR rparen | id | num

07:  FDECL → vtype id lparen ARG rparen lbrace BLOCK RETURN rbrace

08:  ARG → vtype id MOREARGS | ε

09:  MOREARGS → comma vtype id MOREARGS | ε

10:  BLOCK → STMT BLOCK | ε

11:  STMT → VDECL | ASSIGN semi

12:  STMT → if lparen COND rparen lbrace BLOCK rbrace ELSE

13:  STMT → while lparen COND rparen lbrace BLOCK rbrace

14:  COND → COND comp COND | boolstr

15:  ELSE → else lbrace BLOCK rbrace | ε

16:  RETURN → return RHS semi

수정 사항 :

05:  EXPR → EXPR addsub EXPR | EXPR multdiv EXPR

06:  EXPR → lparen EXPR rparen | id | num

* 위 사항은 곱셈을 먼저 해야하는지 덧셈을 먼저해야하는지 모호하기 때문에 아래와 같이 수정한다.
* EXPR -> TERM addsub EXPR | TERM

TERM -> FACTOR multdiv TERM | FACTOR

FACTOR -> lparen EXPR rparen | id | num

14: COND → COND comp COND | boolstr

* 여러 개를 비교할 때 순서에 대해 모호하는 경우가 생겨 바꿔야 한다
* COND -> COND comp BACKCOND | BACKCOND

BACKCOND -> boolstr

수정 후 :

01: CODE → VDECL CODE

02: CODE -> FDECL CODE

03: CODE → ε

04: VDECL → vtype id semi

05: VDECL → vtype ASSIGN semi

06: ASSIGN → id assign RHS

07: RHS → EXPR

08: RHS → literal

09: RHS → character

10: RHS → boolstr

11: EXPR → TERM addsub EXPR

12: EXPR → TERM

13: TERM → FACTOR multdiv TERM

14: TERM → FACTOR

15: FACTOR→ lparen EXPR rparen

16: FACTOR→id

17: FACTOR→num

18: FDECL → vtype id lparen ARG rparen lbrace BLOCK RETURN rbrace

19: ARG → vtype id MOREARGS

20: ARG →ε

21: MOREARGS → comma vtype id MOREARGS

22: MOREARGS → ε

23: BLOCK → STMT BLOCK

24: BLOCK →ε

25: STMT → VDECL

26: STMT →ASSIGN semi

27: STMT → if lparen COND rparen lbrace BLOCK rbrace ELSE

28: STMT → while lparen COND rparen lbrace BLOCK rbrace

29: COND → COND comp BACKCOND

30: COND → BACKCOND

31: BACKCOND → boolstr

32: ELSE → else lbrace BLOCK rbrace

33: ELSE → ε

34: RETURN → return RHS semi

* 1. 절차

주어진 CFG로부터 상태 전이 및 감축 규칙을 파악하여 딕셔너리로 파싱 테이블을 생성한다. 그렇게 입력을 받은 뒤에 토큰들을 확인하여 상태를 파악하고 파싱 테이블에 따라 동작을 달라지게 하였다.

Shift의 경우에는 상태를 스택에 푸시를 하게 되고 Reduce의 과정을 거친다면 viable prefix에서 handle을 파악하여 reduce의 과정을 거친다. Reduce를 한 뒤에 어느 상태로 이동할지 파싱 테이블에서 파악하여 그 상태를 또 다시 스택을 쌓게 된다.

이러한 과정들이 중간에 오류없이 진행이 되고 accept 동작을 거치게 되면 무사히 Syntax analysis가 성공하게 되는 것이다.

* 1. 작동 예

vtype id lparen vtype id rparen lbrace return literal semi rbrace

위와 같은 입력(input.txt와 다른 입력이다.)을 받으면 아래의 결과가 나온다.



1. SLR parsing table

텍스트, 평행, 라인, 번호이(가) 표시된 사진

자동 생성된 설명

