**Cmm的文法:**

E -> prog

// 外部（全局）声明

prog -> extern-declaration | extern-declaration prog

extern-declaration -> declare-stmt | func-definition

func-definition -> type func-declarator compound-stmt

func-declarator -> identifier () | identifier (param-list)

param-list -> param-declaration , param-list | param-declaration

param-declaration -> type identifier | type identifier[expr]

**//程序开始**

stmt-list -> stmt stmt-list | stmt

stmt -> if-stmt | while-stmt | read-stmt | write-stmt | assign-stmt | declare-stmt | compound-stmt

compound-stmt -> { stmt-list } | { }

declare-stmt -> type declarator-list ;

declarator-list -> declarator, declarator-list | declarator

type -> int | real

declarator -> identifier | identifier[expr] | identifier initializer | identifier[expr] initializer | identifier[] initializer

initializer -> = expr | = { initializer-list }

initializer-list -> expr | expr, initializer-list

//if语句

If-stmt -> if ( expr ) compound-stmt more-ifelse

more-ifelse -> ε | **else** else-stmt

else-stmt -> if-stmt | compound-stmt

//while语句

**while-stmt -> while(**expr**) compound-stmt**

//read语句

read-stmt -> read identifier ;

//write语句

write-stmt ->write expr ;

//赋值语句

assign-stmt -> identifier other-assign

other-assign->=expr ; | [number-int]=expr ;

//表达式

expr -> relational-expr

relational-expr -> simple-expr relational-expr-more

relational-expr-more -> ε | comparison-op relational-expr

//比较符

comparison-op -> < | == | <> | > | <= | >=

//加减操作表达式

simple-expr -> term more-term

more-term -> ε | add-op term more-term

//运算符

add-op->+|-

//乘除操作表达式

term->factor more-factor

more-factor->ε | mul-op factor more-factor

//乘除运算符

mul-op -> \* | /

//因子

**factor -> - number | number | identifier more-identifier | (expr)**

**more-identifier ->**ε | [number-int]

number → number-real | number-int