

Updated Syntax Analysis Grammar

Xavier

2025 年 3 月 6 日

声明

此文法仅针对样例 C++ 编写。规定大写字母为首的单词为非终结符，小写字母为首的为终结符，'...' 表示可出现一次或多次，? 表示可出现 0 次或 1 次，被 \$\$ 包裹的部分为一个整体，etc. 表示省略。

1. 主结构

Program \rightarrow Decl \dots (程序由一个或多个声明组成)

2. 声明部分 (Decl)

Decl \rightarrow StructDecl | Type id Suffix (结构体声明或“类型 *id* + 后续”)

VarDecl \rightarrow Type id VarDeclSuffix

2.1 结构体声明 (StructDecl)

StructDecl \rightarrow struct id? { Decl \dots } id? ;

(仅示例，表示结构体内可以继续声明变量、函数等)

2.2 声明后缀 (Suffix)

Suffix \rightarrow (Param) FunctionBody | VarDeclSuffix

这里是关键!

如果遇到 "(", 说明它是函数声明;

否则走 *VarDeclSuffix* 路径处理变量声明。

2.2.1 函数体 (FunctionBody)

FunctionBody \rightarrow CompoundStatement | ; (可以是函数定义或函数原型)

2.2.2 变量声明后缀 (VarDeclSuffix)

VarDeclSuffix \rightarrow = Expression ; | ;

如果下一个符号是“=”，则初始化；否则直接分号结束。

3. 函数参数 (Param)

Param $\rightarrow \epsilon \mid \text{ParamList}$ (参数可为空, 也可有若干个)

ParamList $\rightarrow \text{Type id} (, \text{Type id}) \cdots$ (简化表示, 多个 *type id* 逗号分隔)

4. 语句部分 (Statement)

Statement	\rightarrow CompoundStatement	块语句
	\mid SelectionStatement	选择语句
	\mid IterationStatement	循环语句
	\mid ReturnStatement	返回语句
	\mid ExpressionStatement	表达式
	\mid VarDecl	变量声明语句
	\mid StructDecl	结构体声明语句
	$\mid \text{break}; \mid \text{continue};$	<i>break</i> 语句和 <i>continue</i> 语句

注意! *Expression* 不含分号, *ExpressionStatement* 含有分号

CompoundStatement $\rightarrow \{ \text{Statement} \}$

SelectionStatement $\rightarrow \text{if} (\text{Expression}) \$ \text{CompoundStatement} \mid \text{Statement} \$$
 $\$ \text{else Statement} \mid \text{CompoundStatement} \$?$

IterationStatement $\rightarrow \text{while} (\text{Expression}) \$ \text{Statement} \mid \text{CompoundStatement} \$$

ReturnStatement $\rightarrow \text{return ExpressionStatement}$

DeclInFunction $\rightarrow \text{VarDecl} \mid \text{StructDecl}$

循环语句日后还将支持 *for* 循环

5. 表达式部分 (Expression)

Expression \rightarrow Expression + Term | Expression - Term | Term

Term \rightarrow Term * Factor | Term / Factor | Factor

Factor \rightarrow id | Literal | (Expression)

Literal \rightarrow Numeric | string_literal | char_literal | boolean_literal