根据语法规则可以得出以下的语句是合法的。

module mo {

struct st{

int8 A = 2, B[10 + 2 | 3 ];

string x = "hello";

char z = '\r';

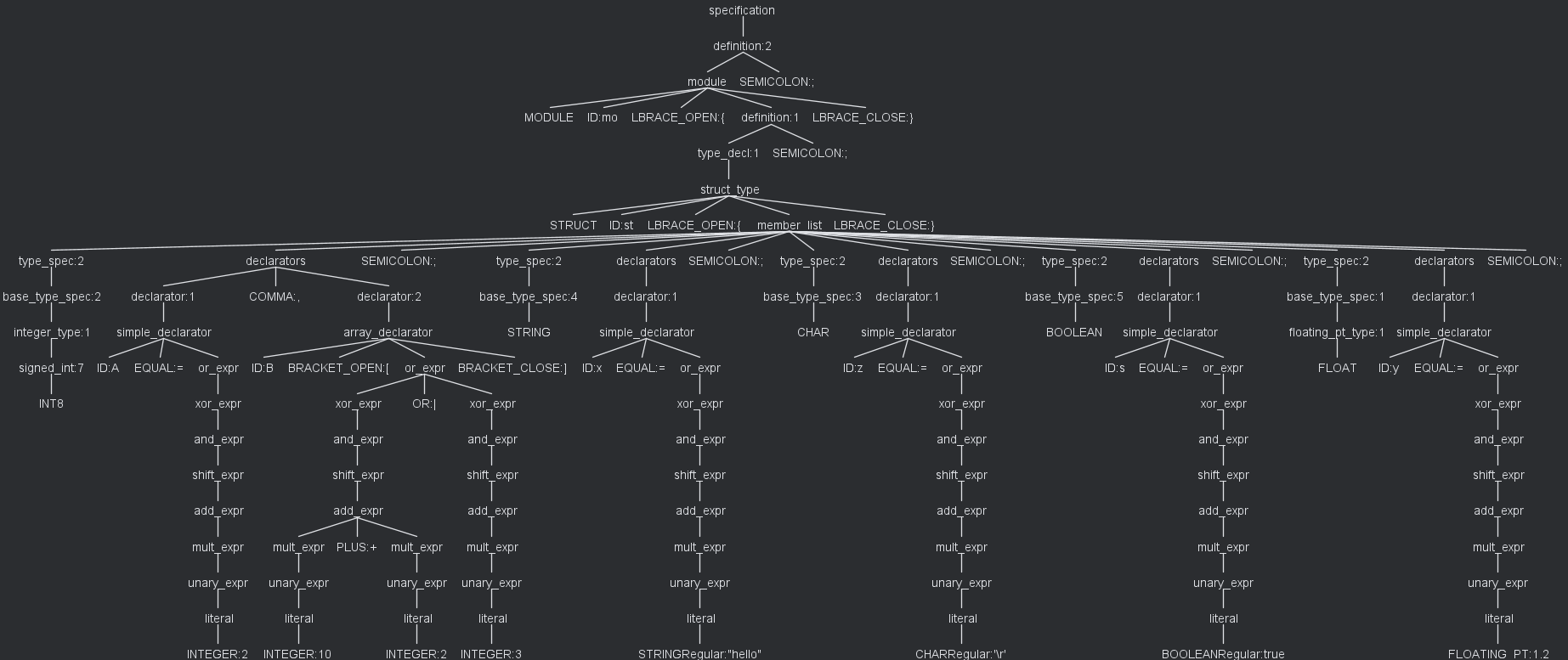
boolean s = true;

float y = 1.2;

};

};

可以将其生成为对应的抽象语法树，见SyntaxOut.doc文件



抽象语法树的树形结构parseTree.png

**MIDL测试**

**1. 词法分析测试（Lexer Tests）**

1.1 关键字识别测试

目的：验证Lexer是否能够正确识别所有MIDL关键字。

测试用例：输入包含部分MIDL关键字的字符串，检查是否被正确识别。

1.2 符号识别测试

目的：验证Lexer是否能够正确识别MIDL语言中的特殊符号。

测试用例：输入包含部分定义的符号（如 {, }, [, ], ;, , 等）的字符串，检查是否被正确识别。

1.3 标识符和字面量识别测试

目的：验证Lexer是否能够正确识别标识符、整数、浮点数、字符和字符串。

测试用例：输入包含各种标识符（如 int8）、整数（如 123）、字符（如 'a'）、字符串（如 "hello"）的字符串，检查是否被正确识别。

**2. 语法分析测试（Parser Tests）**

2.1 模块定义测试

目的：验证Parser是否能够正确解析模块定义。

测试用例：输入类似 module myModule { ... } 的字符串，检查模块定义是否被正确解析。

2.2 结构体定义测试

目的：验证Parser是否能够正确解析结构体定义。

测试用例：输入类似 struct myStruct { ... } 的字符串，检查结构体定义是否被正确解析。

2.3 类型声明测试

目的：验证Parser是否能够正确解析类型声明。

测试用例：输入包含基本类型（如 int8, uint32, float 等）和自定义类型（如 struct myStruct）的声明，检查类型声明是否被正确解析。

2.4 表达式解析测试

目的：验证Parser是否能够正确解析各种表达式。

测试用例：输入包含算术表达式（如 1 + 2 \* 3）检查表达式是否被正确解析。

**3.综合测试（Integrated Tests）**

基于以上两个模块而整合成一个综合测试

3.1 完整模块测试

目的：验证Lexer和Parser是否能够协同工作，正确处理一个完整的MIDL模块。

测试用例：使用提供的测试用例，检查整个模块是否被正确解析。

module mo {

struct st{

int8 A = 2, B[10 + 2 | 3 ];

string x = "hello";

char z = '\r';

boolean s = true;

float y = 1.2;

};

};

测试结果：符合预期，说明该程序能够正确进行词法分析和语法分析。

3.2 错误处理测试

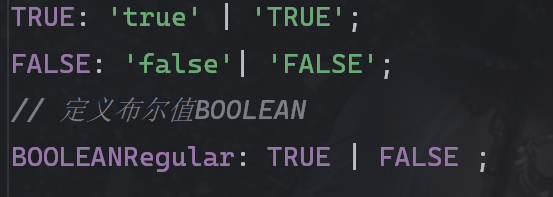
目的：验证Parser在遇到语法错误时是否能够正确报错。

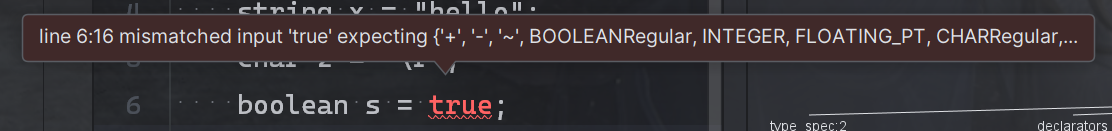
测试用例：输入故意包含语法错误的字符串，检查Parser是否能够识别错误并提供有用的错误信息。

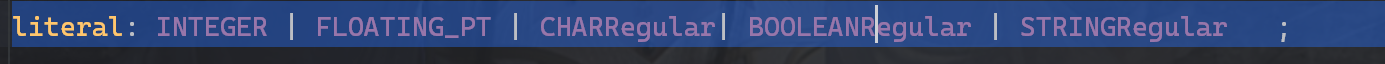


**问题**

在处理boolean的true和false出现一直读不对的问题

在最初，是这么写的词法分析，

报错了

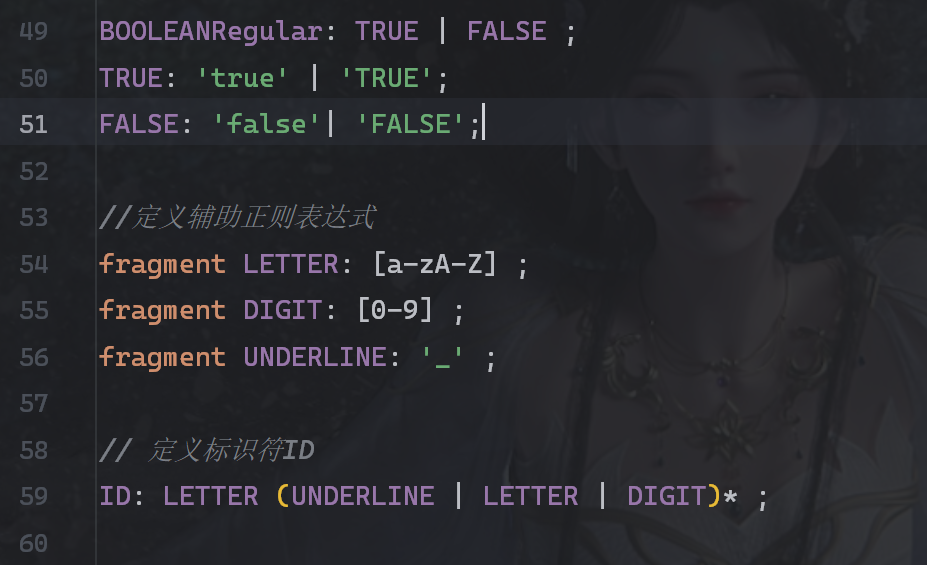


奇怪的是如果把BOOLEANRegular换成TRUE和FALSE就能正确输出了，秉持着探究的精神，我起初以为是因为关键字不能判断？而后我将BOOLEANRegular改成了’true’|’false’发现依旧不对，给我报错。

然后我开始怀疑是位置问题，很离谱，把这段调整顺序后（把TRUE和FALSE挪下来）

然后更神奇的现象发生了：

这三句话放在fragment关键字前面就行了，放在它的后面就不行，



意思就是，现在是正确的，放到56行以后就报错了……神奇的问题

最后怀疑是anltr遇到fragment后把后面所有的都认为不能直接识别一个完整的词法单元token了……因此报错

