编译原理实验一实验报告

161220091 马可欣 3476769672@qq.com

161220092 孟华 dreamanne15@gmail.com

一、实现功能

1. 词法分析，报词法错误

```
Error type A at line 3: Mysterious characters '~'
```

2. 语法分析，正确生成语法树；错误报出错误行号和部分信息（因为...取舍了一部分）

   （生成语法树就不截图了太长了）

```
mh@debian:~/Projects/Compile/Lab1_161220
Error type B at line 4: Missing ']'
Error type B at line 4: Missing '['
Error type B at line 5: Missing ';'
```

3. 注释识别及报错（由于写法问题，如果文件中只有'/*'而没有'*/'程序将不会停止【EOF也被识别为注释】）

```
Error type B at line 6: not allowed single */
```

二、如何编译

1. 编译命令：make　清除可执行文件：make clean
2. 测试命令：（1）Code文件夹下：make test 【测试../Test/test1.cmm】

   或 ./parser ../Test/test1.cmm

   （2）Lab文件夹下：./parser Test/test1.cmm

三、数据结构介绍

```
struct Node {
    char* name; //The name of the Terminal
    enum NodeType nodeType; //Variable:non-ter | Terminal | Num | Type | Id
    int lineNum; int childNum;//child of Num
    char stringValue[300]; //record the value of ID and TYPE
    struct Node* child[8];//Because the most number is 7, so 8 is enough
}
```

四、实验遇到的困难及解决

1. ID和TYPE因为需要输出具体值，所以在建立语法树的时候需要记录具体的值，这个过程我依次犯了三个错：

1. 最一开始的时候，直接令$stringValue = yytext$，但是这样在打印的时候会出现很诡异的错误，一些ID或者TYPE的输出值远远超过它应该有的长度（甚至直接是后面所有的程序）；经过思考，发现字符串赋值并不能直接将地址相等，应该用strcpy进行。因为strcpy需要给预先分配好空间的字符串复制，所以在群里问了一声字符串最大长度...得到的结论是需要动态考虑。**好的，在进行字符串复制的时候，如果超过预留的300长度，则malloc一个对方字符串的的长度，地址复制过去，原地址空间释放。**
2. 然后用strcpy之后发现还是有问题，比如"int main()"，这个ID会记录为"main()"，这是因为，识别到main为ID之后，语法分析并不确定执行哪一条，所以需要识别到LP RP之后才可以进入语法分析的规约步骤，所以此时的yytext已经被更新为"main()"了（**我也不知道为啥？？！！别人都没遇到，就我遇到了><**）。为了解决这个问题，我引入了全局变量string来记录此时的ID，然后在语法分析里引用。
3. 好的，第三个问题来了==，会遇到TYPE ID ID（比如struct Complex x）然后就覆盖掉我的全局变量了……所以我又把单独ID识别的规约规则变回了yytext，然后**加上了分割出ID的操作。**

2. 第二个困难就是写注释的时候，一开始很开心的识别到注释就像处理空格一样什么都不做，然后一想，不对鸭，那怎么报错呢？于是传词法单元给语法，然后越改越乱....最后发现，其实什么都不做就可以了，因为*/会自动被识别为错误……

3. 当然最后一个难点就在于如何加error，一开始一直按照教程上说的error SEMI，但是这样有个问题，如果分号缺失我该怎么报错呢？万一分号缺失然后下一个分号在下一行这报错行号不就错了？于是开始前面匹配，错误部分error，比如Exp error，但是这样同样也有个问题，明明不是错也匹配上了，最后解决办法就是，通过多次调试+测试，找出比较合适的&&不影响正常识别的报错规则。

   报error还有一个问题就是，每抛出一个error，bison会自动调用yyerror打印，如果我想要输出报错信息比如missing ';'，需要在error的函数体中print，然而，程序并不会和你想象中的一样，很有可能是先print后打印yyerror信息，就很奇怪....一气之下让yyerror什么也不做了，就直接在error里面报错，当然....这样就会大幅度减少报错信息（不过还是强于syntax error）。

4. 为报错加了颜色以及报错说明，方便更好的读取。