实验 1 词法分析和语法分析

151220127 吴昕

实现功能：必做和选做全部完成

1. 识别八进制、十六进制，识别指数形式的浮点数。
2. 注释滤除，包括块注释和行注释。
3. 对所有词法错误和语法错误指出错误类型和出错位置。
4. 对没有错误的程序打印语法树。

编译方法：

1. 编译：命令行>make
2. ./parser ../pretest/*.txt，*替换成想要测试的代码编号

make test 可以测试一个样例

3. 清除中间文件：make clean

实验过程：

1. 词法正则表达式

附录 A 已给出大部分词法单元的正则表达式，需要自己编写的只有 INT, FLOAT, ID.

(1) 对于 INT，其包含十进制 DECNUM，八进制 OCT 和十六进制 HEX。分别对这三者写出正则表达式，再将其求并赋给 INT。此外，为了能识别八进制和十六进制的词法错误另设定 OCT_ERROR 和 HEX_ERROR。

当词法分析扫描到八进制和十六进制时，用函数 strtol 将其转换为十进制，类型 INT。

(2) 对于 FLOAT，考虑因素较多，包括其首位是否有正负号，数字与小数点的位置以及是否使用科学计数法以及末位是否有 f。直接赋代码如下：

```
digit [0-9]
FLOAT [+-]?(({digit}*[.]{digit}+)|({digit}+[.]{digit}*))([Ee][+-]?{digit}+)?f?
```

相应的对样例中所出现的 FLOAT 不合法形式有如下定义：

```
FLOAT_ERROR [+-]?((({digit}*[.]{digit}*)?[Ee][+-]?{FLOAT}?)|([.][Ee][+-]?{digit}*))
```

OCT_ERROR, HEX_ERROR, FLOAT_ERROR 使得在词法分析时就能识别其不合法，报错 type A，并准确指出词素。

(3) ID 较为简单，考虑首位不为数字，可以有下划线即可。

2. 优先级和结合性

编写过程中会报错"conflicts: x shift/reduce, reduce/reduce"之类。这是语法冲突导致的，一般 bison 会直接帮我们处理，优先选择移入，在 syntax.output 中可以查看，每个 state 中对于相应输入用[]括号起来的内容表示被忽略的产生式。这些是无关紧要的冲突。

对于与实验相关较为密切的冲突则可通过设定优先级和结合性来避免。参考附录 A 表 A-1 用%left 和%right 设定，先设定的优先级低。

3. 语法分析

语法树的建立是自底向上的，而输出是自顶向下的，所以只能在分析过程中保存语法树，最终检查是否有词法或语法错误，若没有则打印语法树。

(1) 设定 struct 类 TreeNode 表示结点，对每个结点维护类型，属性，所在行号以及孩子数，内置孩子结点。目前实验 1 出现最多孩子结点数为 7 个(Stmt->IF LP Exp RP Stmt ELSE Stmt)。

定义一个容纳 yacc 将要处理的对象的数据类型，这个数据类型是一个 Cunion 对象，在 yacc 文件的第一部分使用%union 声明来定义。定义了 token 以后，可以为它们指定 union 中包含的类型 node。%type<node> 指定非终结符号类型为自定义 node。这就表明，当解析器识别到返回的记号时，它可以认为全局变量 yylval 的名为 node 的成员已经被赋与了有意义的值。于是我们可以以 yylval.node 方式使用它，在 lexical.l 中为每个 node 在语法树上创建一个节点(createNode())。

2. 参照附录 A 文法定义，扫描器 scanner 扫描返回词素，创建节点；语法分析对每个非终结符号及对应产生式设定动作，当没有错误产生时创建一个节点。语法树根节点为 PROGRAM，然后逐级往下，生成语法树。对每个产生式的左部创建树节点，然后根据其产生式链接其孩子结点(addChild())。由于孩子结点数不定，addChild 函数采用变长参数。

4. 语法树的建立和输出

语法树的打印较为简单，用递归的方法，出口为叶子结点。

5. 词法和语法错误

(1) 在 lexical.l 中有对注释和注释嵌套的处理，“//”.*{} 去除单行注释，“/*”{…} 去除多行注释。

①其中<COMMENT>[^*\n]*和<COMMENT>”*”+[^*/\n]* {}这一小段可以忽略，给出的动作也是不作为。是因为有时候写注释有以下习惯，

```
/*
 *
 * … */
```

②<COMMENT><<EOF>>对应还没有匹配到*/就发现文件结束，报错。

③注释(/*和*/)嵌套虽然是语法错误，但是由于词法分析时对块注释进行了处理，因此在出现词法单元*/时可以认为是检测到没有/*的单独的*/，不合法的，Error type B。

(2) 对于语法错误，目前无法保证考虑了所有情况，只能根据给出的样例排查，善用 error。实验过程中发现行末缺;或者语块末缺}的情况比较难处理，对此在定义部分%nonassoc MISSING，MISSING 包括 SEMI, RC 等，然后对于想要处理的 error 语句可以用%prec MISSING 代替。

6. 停止解析

通过使用 yyin 文件指针指向不同的文件，直到所有的文件都被解析。yywrap()这一函数在文件（或输入）的末尾调用，返回 1 来表示解析的结束。因此它可以用来解析多个文件。写在第三段，这就能够解析多个文件。

参考资料：

https://www.ibm.com/developerworks/cn/linux/sdk/lex/

特别鸣谢：

实验中由于对错误恢复不熟悉，对 error 的使用和在规则中放置位置有困难。比如想检测出语句 if () exp)))));这样的错误，用%prec MISSING_RP 会产生冲突。在咨询上一届助教之后修改规则为 Exp error SEMI 即可。在此感谢王慧妍学姐解答！