# Compiler

Experiment one: Lexical Analysis

Name: 朱润身

Student Number: 111220195
Email: runshen.zhu@gmail.com

Date: 30/3/2014

## Environment: Ubuntu 12.04

## Tools: gcc flex

## How to compile

写了一个makefile，在终端中键入“make clean”, 会清除之前的*.o *.yy.c 和可执行文件compiler。在终端键入”make”, 会编译并生成可执行文件compiler

## How to execute it

在终端输入”./compiler”会缺省的对example.txt中的内容做词法分析。如果想对其他内容做词法分析，可以改变example.txt中的内容，也可以另写一个文件，运行时将路径和文件名作为参数传递给compiler就可以。比如”./comilper ./example.txt”. complier会分析当前目录下example.txt文件中内容的词法

## Experiment Content

分别实现了如下种类的词法分析

| | | |
|---|---|---|
| int 十进制整数 | float 十进制浮点数 | bool |
| hex 十六进制数 | oct 八进制数 | if wile do 等保留字 |
| id | whitespace 空白符 | |

同时还有“//”和”/* xxxxxxx */”这两种注释方式

以及不符合词法规范的报错行为

在example.txt中详尽的包含了不同种类的样例，以及各种错误定义的样例。

## Examples

在example.txt中包含了所有的测试用例，参照上面的How to execute it来使用它。共有三部分组成，由”//”注释符号分割。第一部分是所有正确的词法定义，会输出所识别到的词素以及它的类型。第二部分是错误词法的定义，当执行到这部分时会调用err_handle()函数进行一些错误输出。第三部分是用来检测”/*”类的注释，这一部分的内容应该没有任何输出。

## Grammar

沿用了C–的grammar文档

# Highlights

## look_fwd()函数

实验中有一种错误处理比较困难，即“a2.0”这类的。a2.0不符合id的定义，但词法分析会将它拆成id a和float 2.0这两种正确的定义。类似的还有2.2.2，会被处理成2.2和.2两个float (我遵循gcc的识别方式，将.2认为是一个正确的float型)。

为处理这种将一个错误类型拆成两个正确类型的识别方式，我想了两种处理方式，一是详细的定义错误类型，尽量枚举出所有可能的错误方式。二是预读取一个字符，这有点像语法分析。

最终我选择了第二种方式，实现了一个look_fwd()函数。这个函数会在正则匹配成功后调用，预先读取下一个字符，做一些判断以确定当前匹配是否合法。比如当匹配到id时，下一个字符如果是空白符” ”，分号”;”，制表符”\t”，或换行符”\n”，当前匹配才被认为是正确的。否则报错。

## err_handle()函数

类似的同时我实现了一个err_handle()函数，专门用来处理错误，它会输出错误行号和错误字符串。如果look_fwd()函数返回false，或者所有的正则匹配都失败，这个函数会被调用。它将通过input()函数，一个字符一个字符的从流中读取并输出，直到遇到一些结束的标志，比如空白符，制表符之类的。

## 其他

按照要求实现了多种形式的整数及浮点数表示。可以表达八进制、十六进制的数值。

实现了两种格式的注释的识别。 /*....*/和//

其中/*.....*/是通过正则匹配方式识别注释中的内容的。而//则是通过flex的字符处理函数input()实现的，这就有点像err_handle()函数的实现，匹配到//后不停地读取字符但不对他们做处理，直到遇到结束符号'\0'。

# Reference


- http://bbs.csdn.net/topics/80506736 //从这里学到了/*类注释的正则匹配方式
- http://cs.nju.edu.cn/changxu/2%20compiler/projects/%E6%8C%87%E5%AF%BC%E6%94%BB%E7%95%A5%201.pdf //许畅老师班的指导攻略