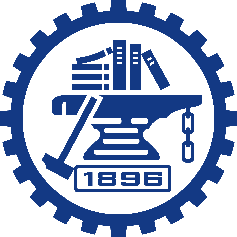
** 词法分析 C:\Users\tangbo\Desktop\学习\基础实践\SJTU PPT\gif\毛体校名（宝蓝色）2.gif**

1. **目标**

本次作业的目标为利用flex编写一个词法分析器，实现对Seal++语言的词法提取。

1. **说明**

本次作业主要利用flex，定义Seal++中的词法格式，从而分析Seal++文件中的符号，将其转化为合适的Seal输出文件，用于进一步的语法分析。本次实验会牵涉到flex、正则表达式[[1]](#footnote-0)等内容，对于flex将有相关参考资料。

1. **flex简介**

flex是lex的一个开源实现（见课本附录A），它能够根据用户定义的正则表达式，对输入文件中的字符串进行匹配，并且对匹配的结果做出相应的处理。这也是词法分析器的基本功能。

flex能够将用户编写的规则文件编译为C源代码（C与C++语法相似，故可以以C++形式继续开发），而编译后的文件可以直接作为库引用。而库代码往往非常繁杂，但是此次作业重点不在编程，故在此次代码设计任务中，其他部分都已经给出，同学只需要编写相应的flex规则文件，也即词法构成规则即可。

flex规则文件的基本结构如下

%{

声明（Declarations）

%}

定义（Definitions）

%%

规则（Rules）

%%

处理函数（User subroutines）

声明和处理函数部分是可选的，可以在其中编写一些辅助用的函数。定义部分也是可选的，但是通常对正则表达式编写会非常有用，例如定义

LOWERCASE [a-z]

简明的定义了小写字母。flex中使用的常见的正则表达式可以查阅课本附录A，在此不再赘述。举例如[xy]，表示字符x或者y。

在flex中最重要的是规则部分，例如

[0-9]+ {// 处理函数}

将会对匹配到的符合[0-9]+的字符串（数字串）做对应的处理函数动作。注意，正则表达式以最长匹配原则，也即如果有[0-9]+和[0-9a-z]+两个正则式，则对于2a这个串，将按照第二个[0-9a-z]+匹配，而不按照一个[0-9]+一个[0-9a-z]+匹配，因为前一种匹配方式更长。

1. **文件说明**

* Little\_lexer.l

flex的规则文件，也即此次作业要编写的文件，目前为空白。

* 测试文件和测试答案

在sealpps文件夹里给了5个样例文件，在test\_results文件夹中有对应的5个词法分析结果文件。可以利用自己编写的词法分析器预先分析，然后与结果对比，结果要求与样例答案完全一致。

* 需要分析的符号等

TYPEID = "Int","Float","String","Bool","Void"

Symbol={"%","~","^","-","\*","/","+","-",";","=",":","|","&","{","}","(",")",",","<",">","%f","%s" ,"%lld","|","&","||","&&"}

'#'为注释符号，其后的数值皆不分析

KEYWORD= "fprintf","while","aafor","if","else","continue","break","return"

1. **如何测试**

首先，在测试之前，请确保机器上安装了flex，具体的，ubuntu机器请运行sudo apt install flex bison来安装。

使用./do.sh进行编译

运行python test.py来测试。

当然也可以单个测试，./little\_lexer < ./sealpps/1.sealpp

1. **处理结果要求**

此次作业要求提交的代码满足如下1个条件：

1. **输出结果**

要求将输入文件代码的每个识别出的单词符号，一行一个输出，格式为

#<单词符号出现行号> <类型> [值]

要求将每个符号都定义相关规则。例如，如果第三行匹配到了一个单词符号为BOOL\_CONST类型，且值为true，则对应行的输出为#3 CONST 1。将所有数值类型的值输出都为#num\_line CONST value。又例如，在第一行匹配到一个单词符号左花括号{，由于其没有值，故输出为#1 {。

1. **如何评分**

作业提交之后，我们将根据同学提交的词法分析器对若干个已有的样本进行分析（与给出的实例样本不完全相同），其中样本可能包含词法错误，包含错误的样本要求输出错误及其行，给出错误原因和出错行为通过，无错误样本要求将输出结果与标准结果比对，完全一致的通过，否则不通过。按照所有样本分析通过率给分（即例如满分10分，通过率0%，给0分，通过率70%，给7分）。

1. **文件提交要求**

仅提交所有源代码文件，最重要的是那个名叫little\_lexer.l的文件，如果同学希望自己用很多别的.h文件或者库，那就需要自己改do.sh中编译的语句。最终在助教端测试的时候，助教会直接运行do.sh来编译得到little\_lexer的二进制文件并用其进行测试。放置在一个名为<学号>的目录下，并且将整个目录打包为<学号>\_<姓名>.tar格式。

1. **提交截止时间**

请同学们在2021.11.14的0:00（2020.11.13的24:00）之前，将结果提交到canvas（迟交结果按0分处理）。

1. 可参考https://www.runoob.com/regexp/regexp-syntax.html [↑](#footnote-ref-0)