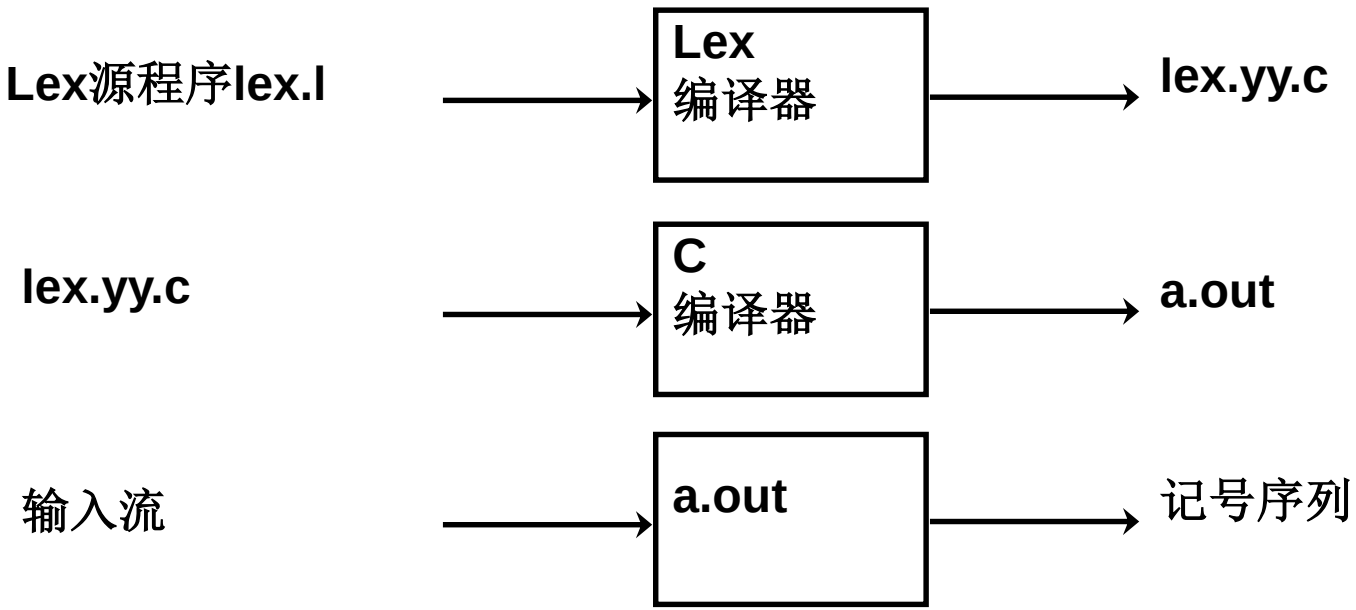
词法分析器

- 用Lex建立词法分析器的步骤



词法分析器

- **Lex程序包括三个部分**
  声明
  %%
  翻译规则
  %%
  辅助过程

- **Lex程序的翻译规则**

p_1	{动作1}
p_2	{动作2}
…	…
p_n	{动作n}

词法分析器

- 例——声明部分

```
%{
/*  常量LT, LE, EQ, NE, GT, GE,
        WHILE, DO, ID, NUMBER, RELOP的定义*/
%}
/* 正则定义 */
delim          [  \t \n ]
ws             {delim}+
letter         [A –Za – z]
digit          [0–9]
id             {letter}({letter}|{digit})*
number         {digit}+(\ .{digit}+)?(E[+\–]?{digit}+)?
```

词法分析器

- 例——翻译规则部分

```
{ws}        {/* 没有动作，也不返回 */}
while       {return (WHILE);}
do          {return (DO);}
{id}        {yylval = install_id ( ); return (ID);}
{number}    {yylval = install_num( );
                                  return (NUMBER);}
" < "       {yylval = LT; return (RELOP);}
 " <= "     {yylval = LE; return (RELOP);}
" = "       {yylval = EQ; return (RELOP);}
" <> " {yylval = NE; return (RELOP);}
" > "       {yylval = GT; return (RELOP);}
" >= " {yylval = GE; return (RELOP);}
```

词法分析器

- 例——辅助过程部分

```
installId ( ) {
        /* 把词法单元装入符号表并返回指针。
        yytext指向该词法单元的第一个字符，
        yyleng给出的它的长度          */
}

installNum ( ) {
        /* 类似上面的过程，但词法单元不是标识符而
    是数 */
}
```

词法分析器

- Examples
  - tmp.l
    - (a|b)*ab
    - (00|11|(01|10)(00|11)*(01|10))*
    - z{2,4}[^z\n]*
  - Commands:
    - lex –o tmp.c tmp.l
    - gcc –ll tmp.c –o tmp
    - ./tmp

词法分析器

- 可以用到的正则表达式
  - http://dinosaur.compilertools.net/flex/manpage.html
    - PATTERNS
      - 例：
        - » [xyz]
          - a "character class"; in this case, the pattern matches either an 'x', a 'y', or a 'z‘
        - » r{2,5}
          - anywhere from two to five r's
        - » ……

词法分析器

- Homework#1
  - prog.txt
    - int asd = 0;
    - int bc = 10;
    - while (asd < bc)
    - {
    - if(bc - asd < 2)
    - cout<<"they are close."<<endl;
    - asd = asd + 1;
    - }
  - 识别出prog.txt中的各个关键词、()、{}、各个变量、关系运算符、加减乘除运算符、标点符号、流操作符，并输出其相应类型和内容到屏幕。

词法分析器

- Homework#2
  - 从一句英文语句中识别出网址和Email地址并打印它们

词法分析器

- Homework#3
  - 给出下列正则表达式的有限自动机
    - $(1|0)^*00$
    - $1^*01^*01^*01^*$