北京理工大學

词法分析实验

班　　级：07111505

姓　　名：徐宇恒

学　　号：1120151839

目录

一. 实验目的

在已有的 BITMiniCC 框架的基础上，实现词法分析器，加深对词法分析的理解。实现对输入 C 语言源文件进行词法分析得到属性字流。

二. 实验内容

以 C 语言为源语言，java 语言为宿主语言，构建 C 语言的词法分析器，对于任一给定 C 语言源程序，输出 XML 格式的属性字流。

三. 实验环境

操作系统：Windows10 Pro

程序设计语言：C++

IDE：Visual Studio 2017

四. 实验过程

4.1. 设计思路

4.1.1. C 语言词法元素分类：

- 关键字：C 语言关键字共有 32 个，根据关键字的作用可将其分为四类
    - 数据类型关键字:

```
char, double, enum, float, int, long, short, signed,
struct, union, unsigned, void
```

    - 控制语句关键字:

```
for, do, while, break, continue, if, else, goto, switch,
case, default, return
```

- 存储类型关键字:

- 其他关键字

```
Const ,sizeof, typedef, volatile
```

- 标识符：C 语言规定，合法额表示付必须由英文字母或下划线开头，是字母数字和下划线的序列。
- 操作符：

```
“+”，“-”，“++”，“--”，“+=”，“-=”，“*”，“/”，“%”，“^”，“=”，“!”，“*=”，“/=”，“%=”，“^=”，“==”，“!=”，“&”，“&&”，“&=”，“|”，“|=”，“||”，“<”，“<=”，“<<”，“<<=”，“>”，“>=”，“>>”，“>>=”，“~”
```

- 分隔符：

```
“{”“}”“[”“]”“(”“)”“,”“;”“.”“:”“?”
```

- 常量：常量的种类复杂多样，本实验仅实现以下几类。
  - 八进制整数及浮点数：0 开头，且仅有 0-7 构成的序列
  - 十进制整数及浮点数： 1-9 开头，仅有 0-9 构成的序列
  - 十六进制整数及浮点数：0X 开头，仅有 0-9、a-f 构成的序列
  - 八、十、十六进制科学记数法表示
  - 字符以及字符串（含转义字符）
  - 其余均设置为非法

4.1.2. 自动机设计思路：

- 先分别实现能够识别各类词法的自动机，再加上统一的起始状态进行连接。

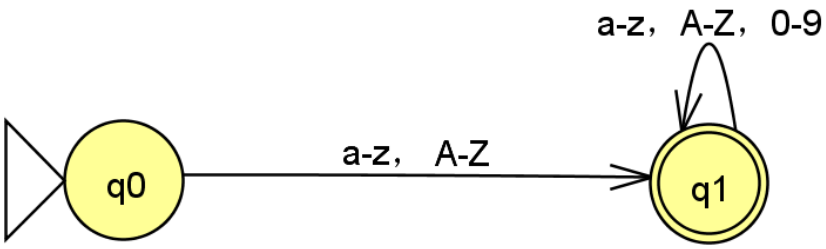
4.1.3. 程序设计思路：

```
int state = 0;          //状态；
char str;
while(1)
{
   str = nextchar();     //自动机读取一个字符，开始识别
   switch(state)         //进行状态转换
   {
      case 0: switch(str) //初始状态，进入不同的类别
      {
         case 'a'...'z':state = 1;break;
         case '0'...'9':state = 3;break;
         case '=':state = 5;break;
         case '*':state = 6;break;
         ......
         default:state = 255;   //未能正确识别的状态
      }break；
      case 1:while(str == letter||number)    //状态 1 是关键字
      str = nextchar();state = 2;break;
      case 2: return;                    //状态 2 是标识符

      ......

      default: state = 255;
   }
}
```

4. 2. 设计自动机

4.2.1. 识别关键字和标识符的自动机

- 由于关键字比标识符的范围小的多，且完全包含在便师傅的范围内。所以只要画出表示付的自动机，在此基础上进行穷举即可得到关键字。
- C 语言规定，合法额表示付必须由英文字母或下划线开头，是字母数字和下划线的序列。
- 图示如下：

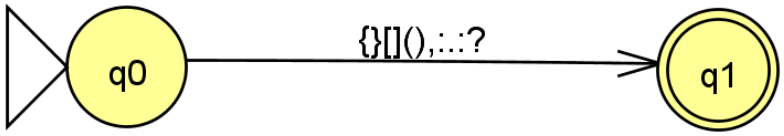


4.2.2. 识别分割符的自动机

- C 语言常用的分割符有：

```
“{” “}” “[” “]” “(” “)” “,” “;” “.” “:” “?”
```

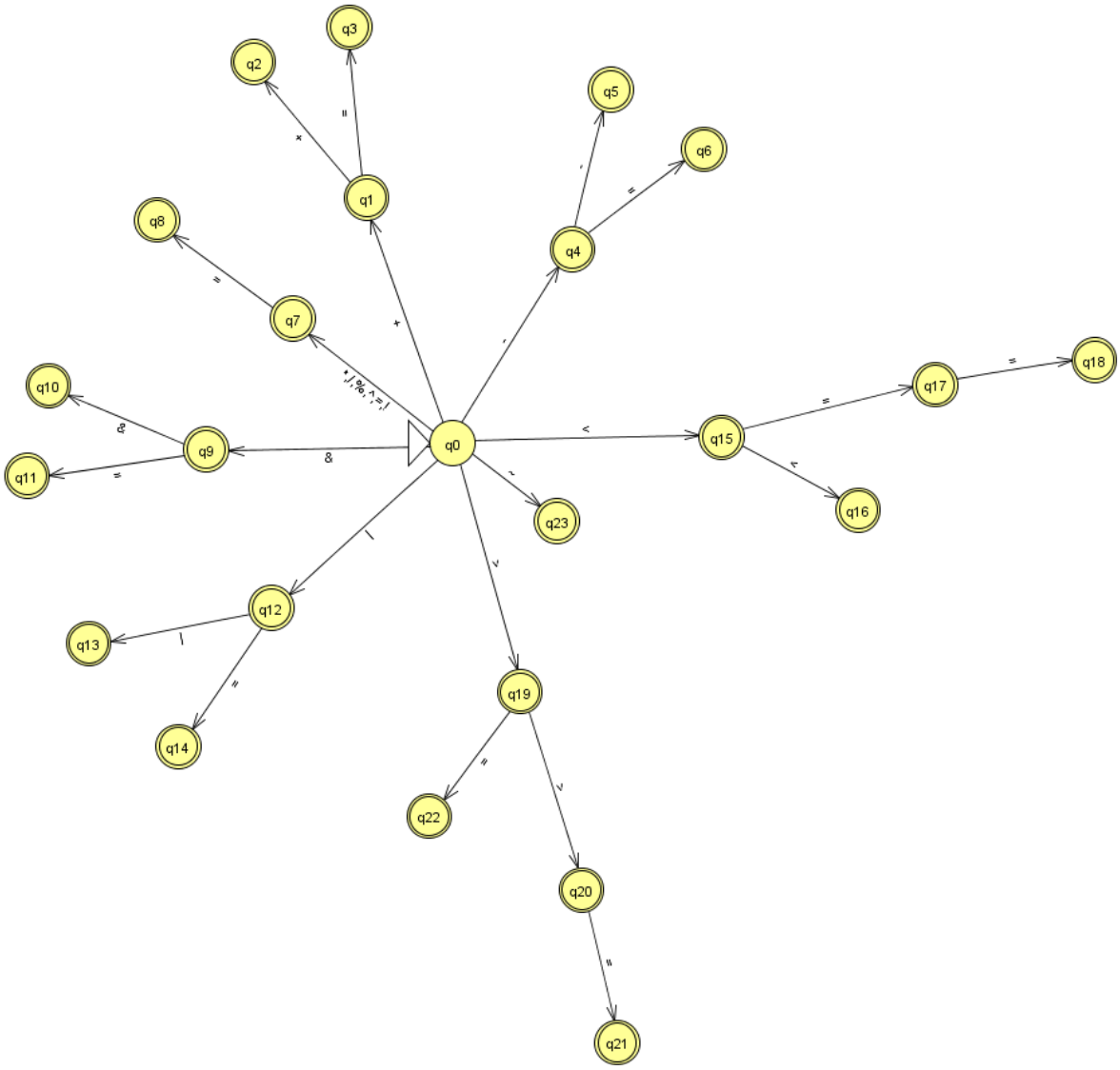
- 图示如下：



4.2.3. 识别操作符的自动机

- 操作符种类多样，且含有多种组合，且都为接受态，所以在最后输出是可归为一个状态输出，减少判断量。

- 图示如下

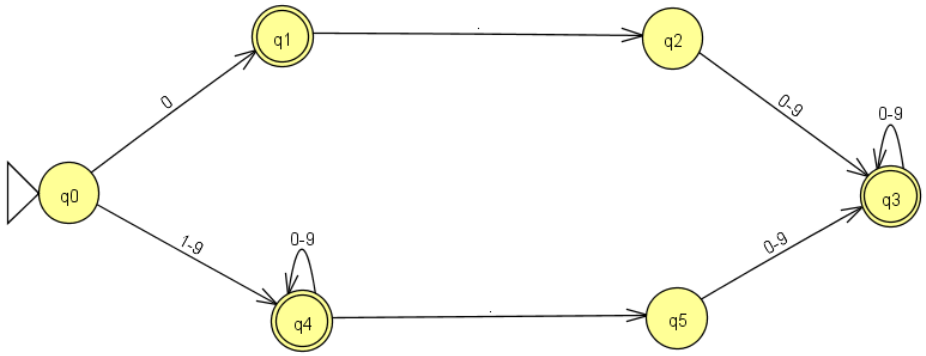


4. 2. 4. 识别常量的自动机

➢ 识别十进制整数以及浮点数的自动机：

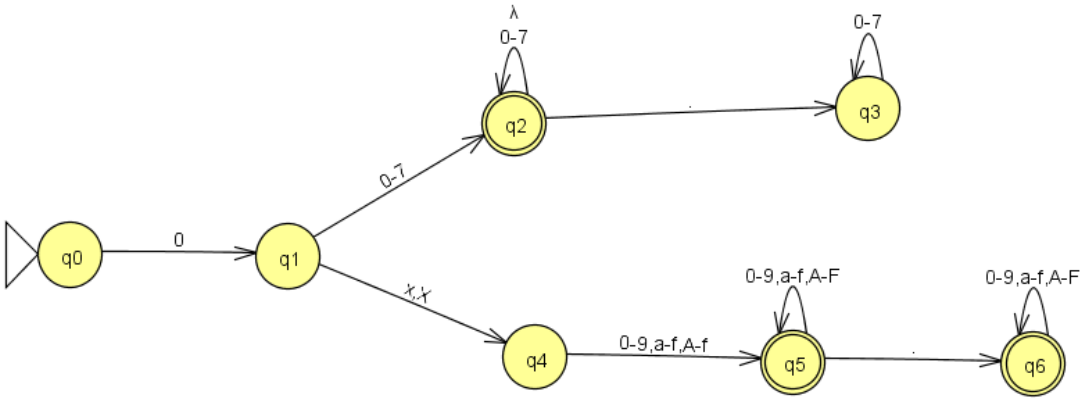
- 十进制中，若首位为 0，则是小数，若首位非 0，则可能是整数也可能是小数。

- 图示如下：



➢ 识别八进制和十六进制整数以及浮点数的自动机：

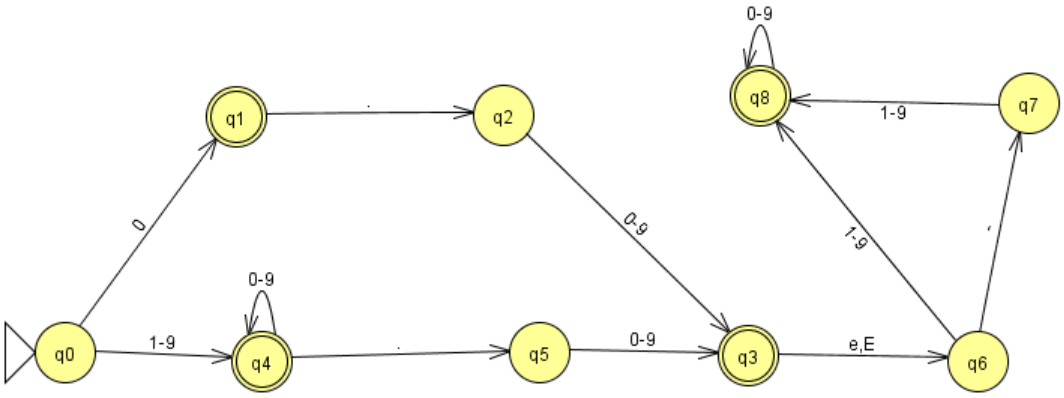
- 八进制与十六进制与十进制的思路相同，需要注意的是八进制的起始字符为 0，十六进制的起始为 0x。8 进制所能识别的数字为 0-7，十六进制能识别 0-9 以及 a-f。
- 图示如下：



➢ 识别科学记数法的自动机：

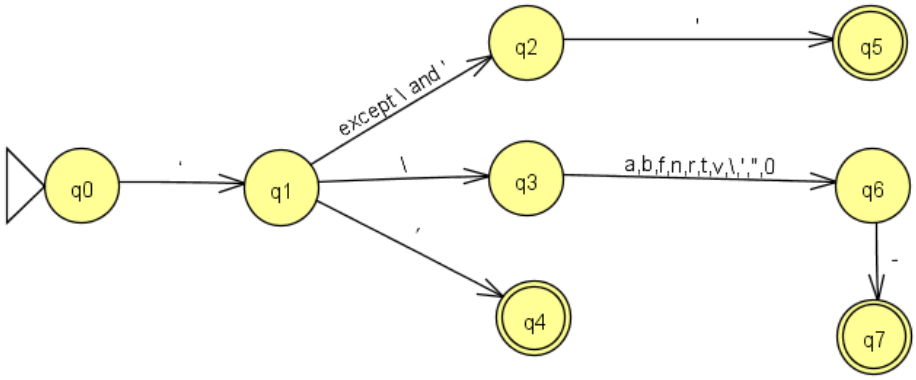
- 注意科学计数器由两部分，字符 e 之前的是底数，e 之后的是阶码。底数和阶码都可正可负。

- 图示如下：



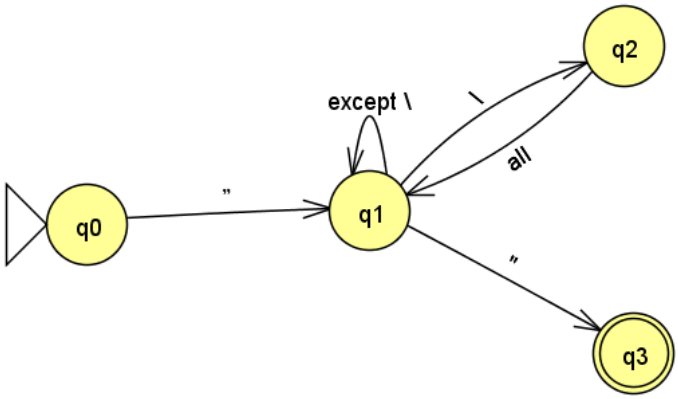
➢ 识别字符常量的自动机（含转义字符）：

- 字符常量起始与结尾都是单引号，中间是任意字符，其中转义字符需要单独区分。
- 图示如下：

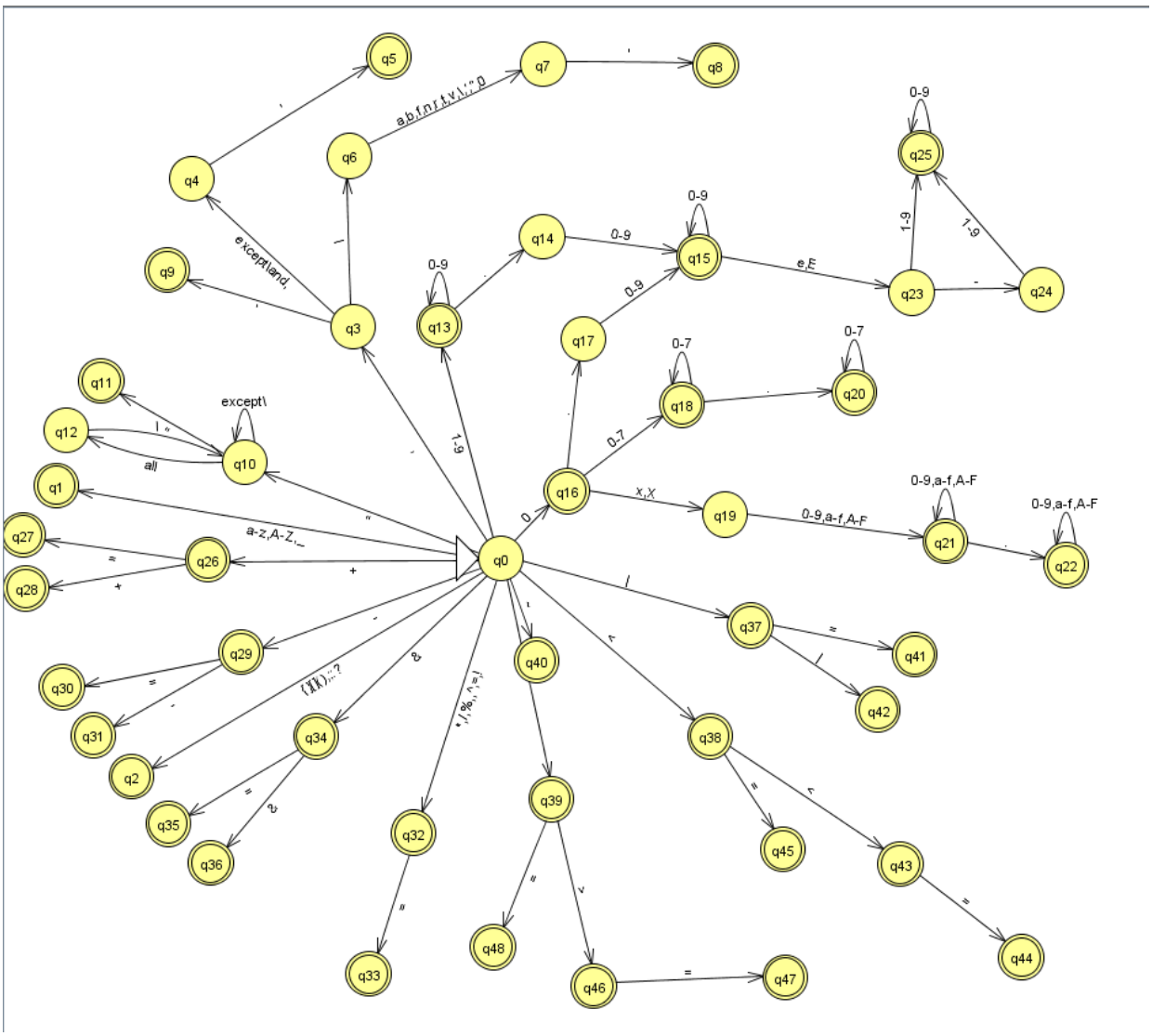


➢ 识别字符串常量的自动机：

- 字符串常量前期是和结束均为双引号。中间可以是任意字符序列，但穿衣字符需要单独考虑。
- 图示如下：


q0
"
q1
except \
\
all
"
q2
q3

4. 3. 词法分析自动机


q0
q1
q2
q3
q4
q5
q6
q7
q8
q9
q10
q11
q12
q13
q14
q15
q16
q17
q18
q19
q20
q21
q22
q23
q24
q25
q26
q27
q28
q29
q30
q31
q32
q33
q34
q35
q36
q37
q38
q39
q40
q41
q42
q43
q44
q45
q46
q47
q48
except\
except\end
a-z,A-Z,_
0-9
1-9
0-7
x,X
e,E
0-9,a-f,A-F
a,b,f,n,r,t,v,\,',",0

五. 实验结果

5.1. 输入的 test.c 为文件

test.pp.c

```c
int main()
{
    int a = 2.6, b = 1e-7, j = 5;
    char s = 'u';
    char *str = "Scanner";
    printf("HelloWorld\n");
    while(j--);
    return 0;
}
```

5.2. 经过预处理和词法分析的 xml 文件（节选）

```
<project name="test.c">
  <tokens>
    <token>
      <number>1</number>
      <value>int</value>
      <type>keyword</type>
      <line>1</line>
      <valid>true</valid>
    </token>
    <token>
      <number>2</number>
      <value>main</value>
      <type>identifier</type>
```

```
    <line>1</line>
    <valid>true</valid>
  </token>
  <token>
    <number>3</number>
    <value>(</value>
    <type>separator</type>
    <line>1</line>
    <valid>true</valid>
  </token>
  <token>
    <number>4</number>
    <value>)</value>
    <type>separator</type>
    <line>1</line>
    <valid>true</valid>
  </token>
```

六. 心得体会

通过本次实验，我对于词法分析器有了初步的认识。

首先在设计程序的思路上，采用先局部后整体的方式，将五个主要的自动机设计出来，在进行合并。好处是复杂的自动机被拆解成了容易实现的小型自动机。缺点是对于整体的把握很不成分，尤其是在模块有交集的地方，是否应该合并，怎么合并都成了伤脑筋的问题。所以在设计的时候应该先把整个框架设计好，对于整体有一定的把握之后，再分成各个模块进行操作。

其次，最大的体会就是用 C 语言编写代码实在是太痛苦了，C 不像 Java 这类语言，对于接口有着严格且明确的定义。尤其是传参时对 C 的理解不够深入，能正

常传递参数，但是实际上却不能够被调用的程序所接受，编译成功但程序还是莫名崩掉，调了两天 bug 才勉勉强强能正常工作。

Ps：逐渐体会到当年为了千万码农写编译器的大师们为什么都掉光了头发。。。