第一阶段报告

20184556 陈然

任务：1符号表的数据结构设计

2 四元式生成

3 目标代码生成

已做：文法

文法如下：

**VARIABLE ->var ID\_SEQUENCE ： TYPE ；{ID\_SEQUENCE ： TYPE ；}**

**ID\_SEQUENCE ->id ARRAY {，id ARRAY }**

**ARRAY->e|[cons]**

**TYPE ->integer | real | char**

**ANOUNCE\_FUNCTION->var id(id :TYPE {,id：TYPE }):TYPE{FUN\_SENTENCE}**

**FUN\_SENTENCE->begin SEN\_SEQUENCE|CONDITION|CIRCLE {；SEN\_SEQUENCE | CONDITION | CIRCLE} RETURN end**

**RETURN->return FACTOR;**

**FUNCTION->id(id{,id})**

**COM\_SENTENCE -> begin SEN\_SEQUENCE|CONDITION|CIRCLE|FUNCTION {；SEN\_SEQUENCE | CONDITION | CIRCLE|FUNCTION} end**

**SEN\_SEQUENCE -> EVA\_SENTENCE {；EVA\_SENTENCE }**

**CONDITION -> if BOOL then SEN\_SEQUENCE else SEN\_SEQUENCE**

**CIRCLE -> while BOOL do COM\_SENTENCE**

**EVA\_SENTENCE -> id ARRAY := EXPRESSION**

**EXPRESSION -> EXPRESSION + TERM | EXPRESSION - TERM | TERM**

**TERM -> TERM \* FACTOR | TERM / FACTOR | FACTOR**

**FACTOR -> id ARRAY | cons | ( EXPRESSION )**

**BOOL -> FACTOR <|>|@ FACTOR**

**其中：id为标识符，cons为常数。**

已做：符号表

（1） 字符数组：char test[1000]

（2） 关键字表：

|  |  |
| --- | --- |
| 关键字 | 编码 |
| program | 0 |
| var | 1 |
| integer | 2 |
| real | 3 |
| char | 4 |
| begin | 5 |
| end | 6 |
| if | 7 |
| then | 8 |
| else | 9 |
| while | 10 |
| do | 11 |
| ifend | 12 |
| return | 13 |

界符表：

|  |  |  |  |
| --- | --- | --- | --- |
| 界符 | 编码 | 界符 | 编码 |
| . | 1 | ( | 10 |
| : | 2 | ) | 11 |
| ; | 3 | < | 12 |
| = | 4 | > | 13 |
| \* | 5 | @ | 14 |
| / | 6 | ( | 15 |
| + | 7 | ) | 16 |
| - | 8 | { | 17 |
| . | 9 | } | 18 |

Token序列结构体

struct Token

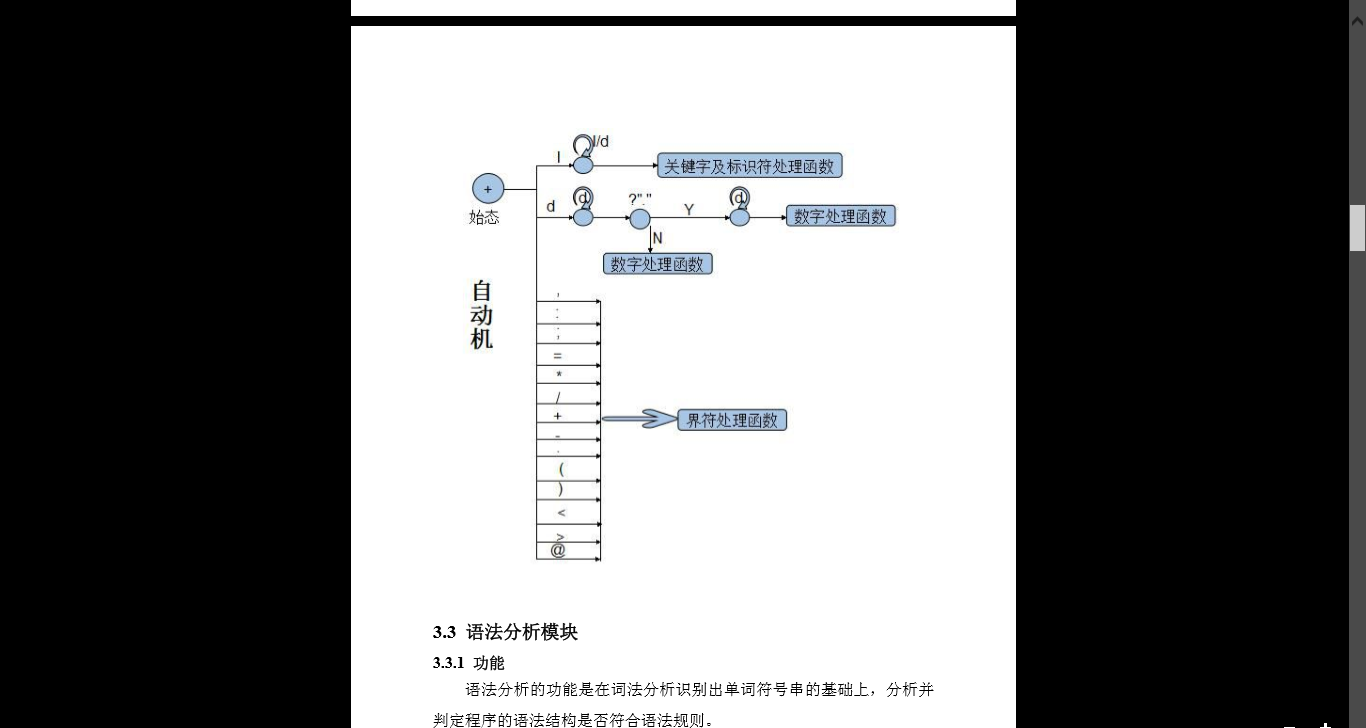
{

char content[10];//用来存放Token类型

int code;//用来存放表中位置

}token[200];//TOKEN序列数组，用来存放生成的token

利用自动机扫描，每一个单词首字符为字母的时候只有两种情况，不是标识符，就是关键字，所以调用关键字及标识符处理函数以生成相应的 token，而单词的首字母为数字的时候只能为数字常量，此时则调用数字处理函数生成相应的 token，而其他的情况，单词只能是界符，此时应该调用界符处理函数。具体自动机如下图所示：



已做四元式生成

在四元式的编写当中定义了四元式的格式，并且对标识符，关键字，等必要的符号写出相关的四元式生成函数如：

void QUATFIRST();

void QUATJIA();

void QUATJIAN();

void QUATCHENGE();

void QUATCHU();

void QUATFUZHI(string len,int type);

void QUATLAST();

void QUATBEGIN();

void QUATBOOL();

void QUATTHEN();

void QUATELSE();

void QUATIFEND();

void QUATWHILE();

void QUATDO();

void QUATWHILEEND();

void QUATHANSHUSHENGMING(string str1,string str2);

void QUATDIAOYONG(string str);

void QUATRETURN();

### 3.5.3算法

以具有概括性的四元式生成的函数为例：

void QUATFIRST();

void QUATJIA();

代码：

void QUATFIRST()//生成第一个四元式

{

string temp="";

siyuanshi[xuhao]="QT["+int2str(xuhao)+"]:(program,"+"I0"+","+"\_"+","+"\_"+")";

siyuanshi1[xuhao1]=temp+"(program,"+"I0"+","+"\_"+","+"\_"+")";

xuhao1++;

xuhao++;

}

void QUATJIA()//加号生成四元式

{

temp1=s.Pop();

temp2=s.Pop();

string\* p=new string;

\*p="t"+int2str(jilu);

siyuanshi[xuhao]="QT["+int2str(xuhao)+"]:(+,"+temp2+","+temp1+","+\*p+")";

siyuanshi1[xuhao1]="(+,"+temp2+","+temp1+","+\*p+")";

s3.Push(\*p);

s.Push(\*p);

xuhao1++;

xuhao++;

}

在代码中可以看到对program的特殊关键字的进行识别生成四元式。

在普通的如：加、减、乘、除这样的四元式的生成函数中一般应用到：双栈结构（对表的内容的正序，四元式表等相关的必要结构）

四元式表代码：

int xuhao1=0;

string siyuanshi1[100];

对于栈及其相关操作函数的代码：

class Stack

{

public:

Stack();

void Push(string b);

string Pop();

void Clear();

string Peek() const;

bool isEmpty() const;

private:

string a[100];

int Top;

};

Stack::Stack()

{

Top=-1;

}

void Stack::Push(string b)

{

if(Top==99)

return;

Top++;

a[Top]=b;

}

string Stack::Pop()

{

string element;

//如果栈空，则返回0；

if(Top==-1)

return 0;

element=a[Top];

Top--;

return element;

}

void Stack::Clear()

{

Top=-1;//直接置栈顶指针，没有清空栈

}

string Stack::Peek()const//取栈顶元素

{

//如果栈为空，则返回0；

if(Top==-1)

return 0;

return a[Top];

}

bool Stack::isEmpty() const

{

if(Top==-1)

return true;

else

return false;

}

遇到问题：接口

解决：先把接口商量好在分开完成任务

还有目标代码生成没有做