编号：

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 实习 | 一 | 二 | 三 | 四 | 五 | 六 | 七 | 八 | 九 | 十 | 总评 | 教师签名 |
| 成绩 |  |  |  |  |  |  |  |  |  |  |  |  |

武汉大学计算机学院

**《编译原理》课程**

词法分析

实习报告

班 序 号： CS42

实习题目： 词法分析

专业（班）： 计算机科学与技术7班

学生学号： 2018302100026

学生姓名： 龙 晓 怡

任课教师： 杜 卓 敏

2021 年 6 月 9 日

**目录**

[1 语言形式化描述 2](#_Toc74127215)

[1.1根据词法规则写出正规文法 2](#_Toc74127216)

[1.2改进文法（非左递归描述） 3](#_Toc74127217)

[2 单词编码表 5](#_Toc74127218)

[3 状态转换图 5](#_Toc74127219)

[4 词法分析算法 6](#_Toc74127220)

[4.1 算法流程 6](#_Toc74127221)

[4.2关键变量定义 6](#_Toc74127222)

[4.3关键函数定义 7](#_Toc74127223)

[4.3.1查找保留字int reserve(const std::string &) 7](#_Toc74127224)

[4.3.2词法分析器DPair LexAnalyze() 7](#_Toc74127225)

[4.3.3报错函数void error(int type) 10](#_Toc74127226)

[4.3.4输出词法分析结果函数void output(const DPair binseq) 10](#_Toc74127227)

[5 测试计划 10](#_Toc74127228)

[5.1正确测试用例 11](#_Toc74127229)

[5.2错误测试用例 11](#_Toc74127230)

[5.3 实验结果 12](#_Toc74127231)

[5.3.1词法分析result.dyd输出结果 12](#_Toc74127232)

[5.3.2错误文件error.err输出 12](#_Toc74127233)

# 1 语言形式化描述

## 1.1根据词法规则写出正规文法

<程序>→<分程序>

<分程序>→begin <说明语句表>；<执行语句表> end

<说明语句表>→<说明语句>│<说明语句表> ；<说明语句>

<说明语句>→<变量说明>│<函数说明>

<变量说明>→integer <变量>

<变量>→<标识符>

<标识符>→<字母>│<标识符><字母>│ <标识符><数字>

<字母>→a│b│c│d│e│f│g│h│i│j│k│l│m│n│o │p│q │r│s│t│u│v│w│x│y│z

<数字>→0│1│2│3│4│5│6│7│8│9

<函数说明>→integer function <标识符>（<参数>）；<函数体>

<参数>→<变量>

<函数体>→begin <说明语句表>；<执行语句表> end

<执行语句表>→<执行语句>│<执行语句表>；<执行语句>

<执行语句>→<读语句>│<写语句>│<赋值语句>│<条件语句>

<读语句>→read(<变量>)

<写语句>→write(<变量>)

<赋值语句>→<变量>:=<算术表达式>

<算术表达式>→<算术表达式>-<项>│<项>

<项>→<项>\*<因子>│<因子>

<因子>→<变量>│<常数>│<函数调用>

<常数>→<无符号整数>

<无符号整数>→<数字>│<无符号整数><数字>

<条件语句>→if<条件表达式>then<执行语句>else <执行语句>

<条件表达式>→<算术表达式><关系运算符><算术表达式>

<关系运算符> →<│<=│>│>=│=│<>

## 1.2改进文法（非左递归描述）

改进文法G'如下：

S→begin A B end

A→C;A'

A'→C;A'|ε

C→integer D

D→<标识符> | function <标识符>（<标识符>）；E

E→begin A B end

B→FB'

B'→;FB'|ε

F→read(<标识符>)│write(<标识符>)│G│H

G→<标识符>:=I

I→JI'

I'→-JI'|ε

J→KJ'

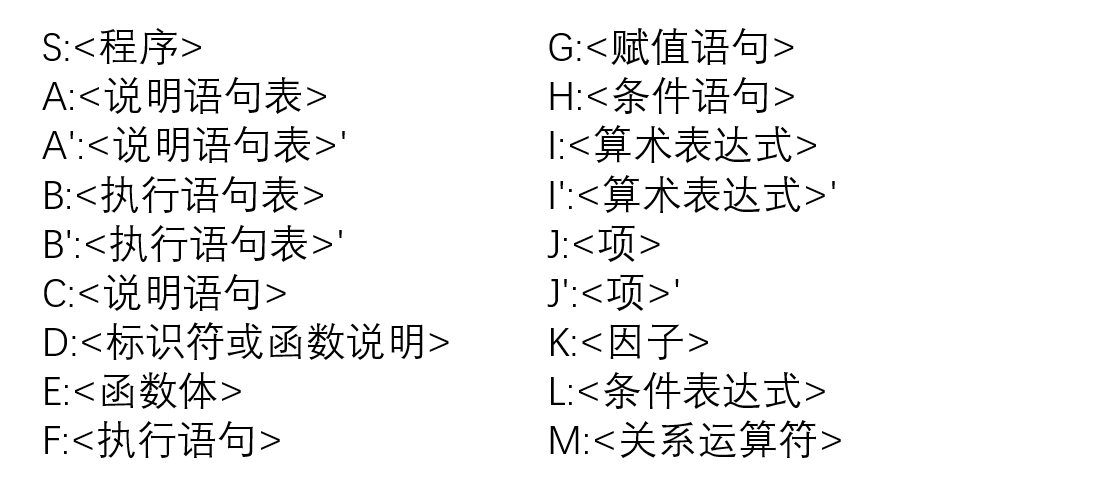
J'→\*KJ' | ε

K→<标识符>│<常数>│<函数调用>

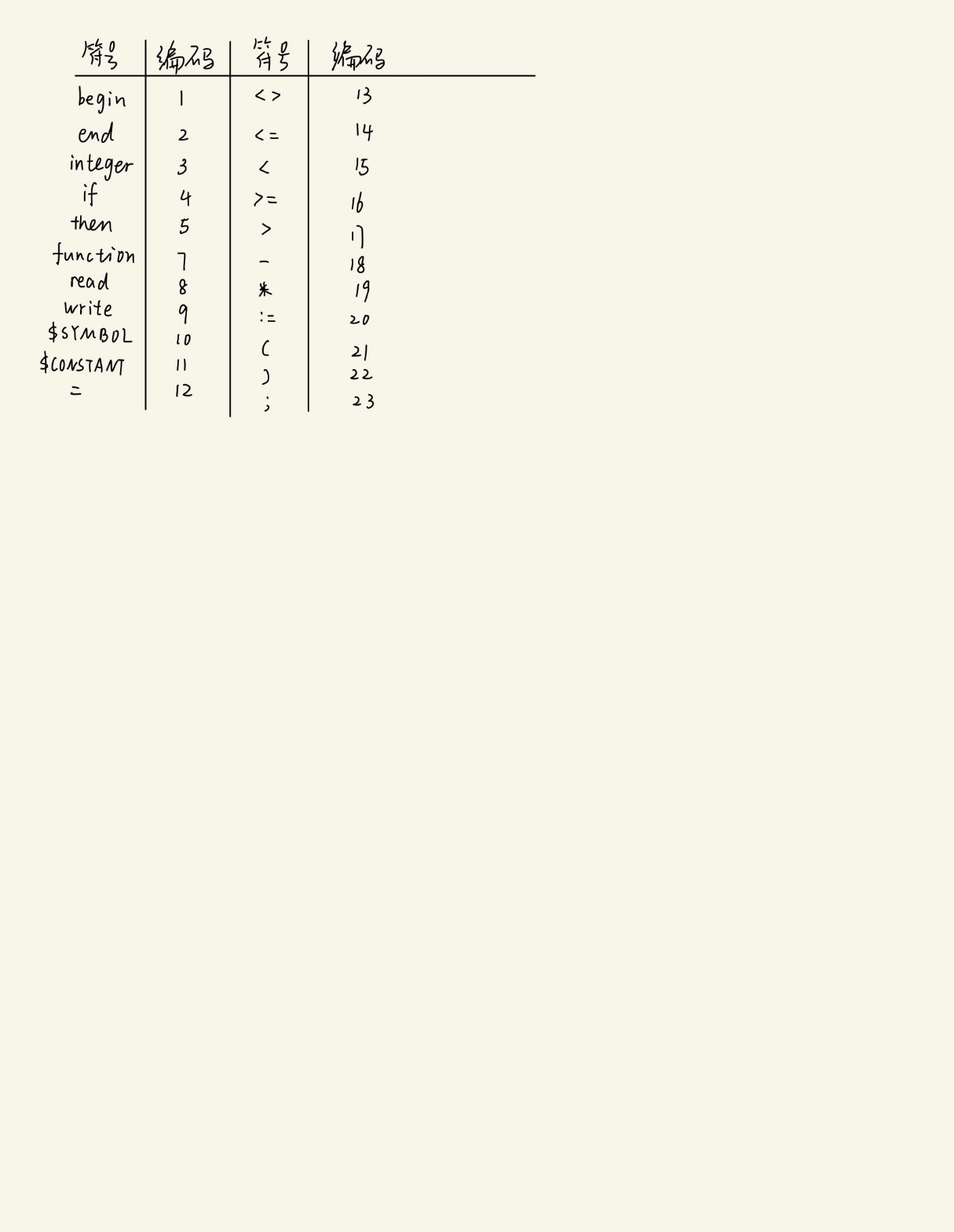
H→if L then F else F

L→IMI

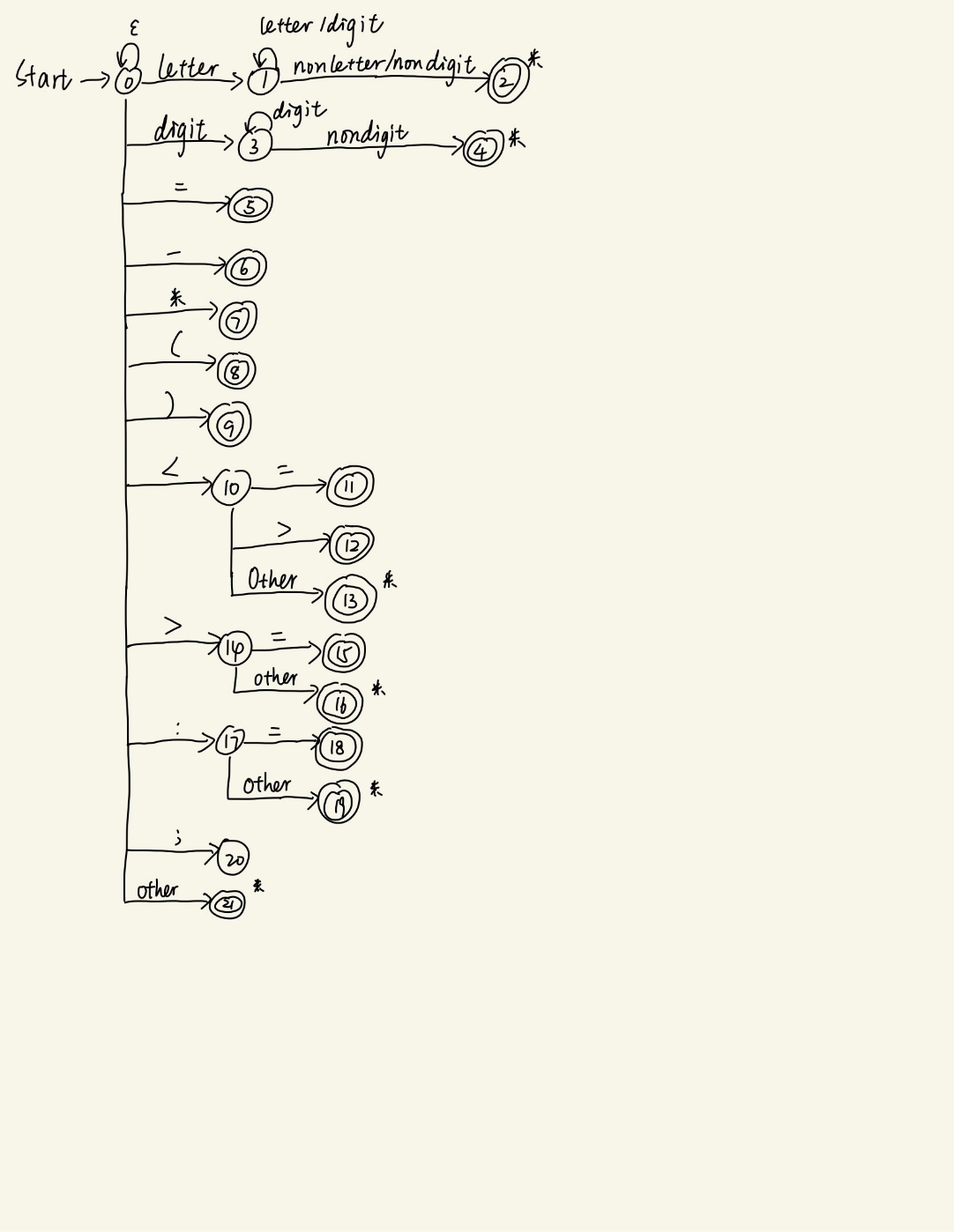
M→<│<=│>│>=│=│<>

其中：

# 2 单词编码表

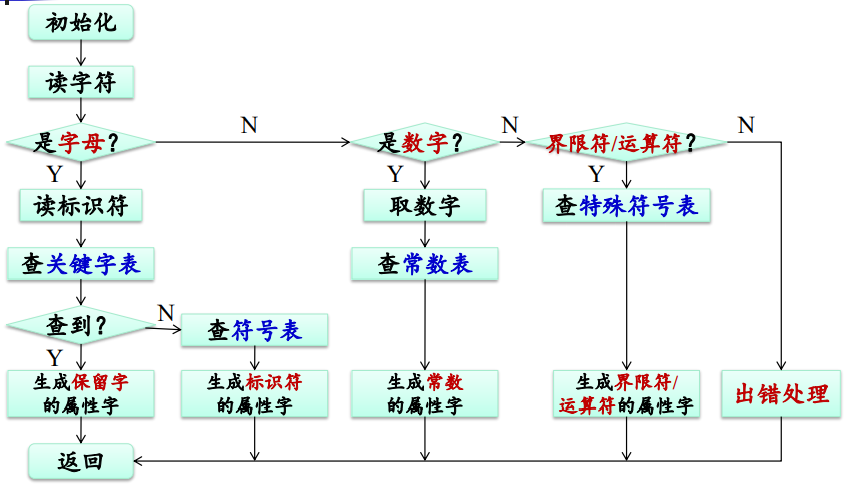
单词编码采用一类一码、一符一码

# 3 状态转换图

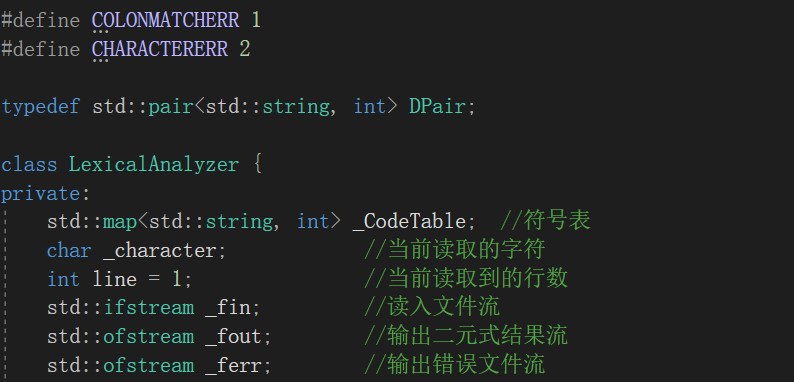
将初始状态合并为唯一的初态

# 4 词法分析算法

## 4.1 算法流程



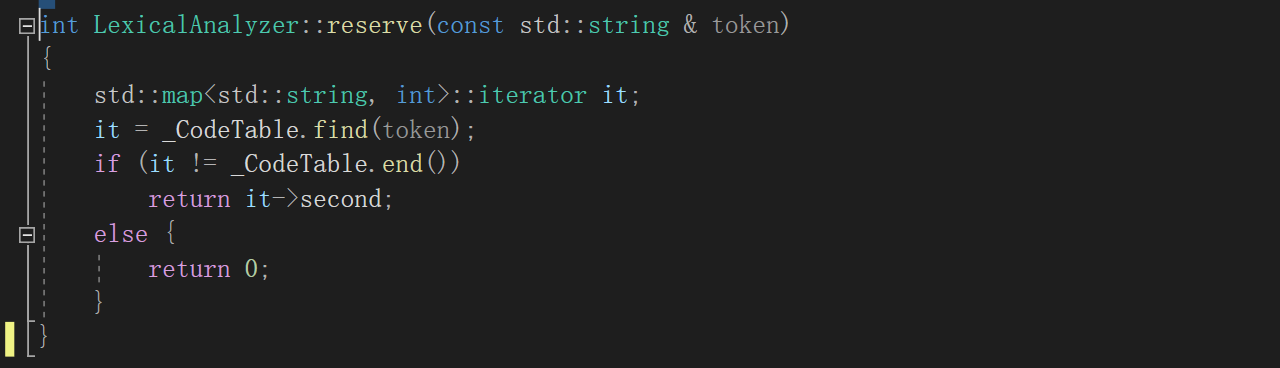
## 4.2关键变量定义



## 4.3关键函数定义

### 4.3.1查找保留字int reserve(const std::string &)

若查找成功，返回种别码

若查找不成功，返回0，即为标识符

### 4.3.2词法分析器DPair LexAnalyze()

该词法分析器完成的主要功能

1）跳过空白。即对于空格，\t，\n字符，需要自动跳过；

2）判断读入的字符（或字符串）是否为某类词法单元；

3）通过若干判断分支，最终得到的返回值即为识别出的一个词法单元.

#### 预处理

#### 分析标识符

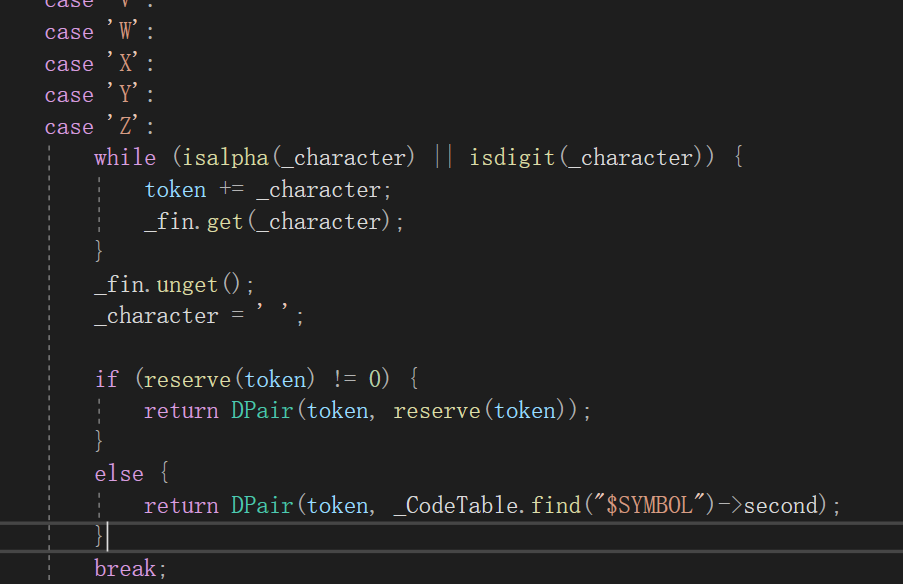
程序到达这里的状态是：当前字符一定是英文字符，且当前字符的前一个字符一定不是英文字符

标识符包含关键字和变量名

注意：

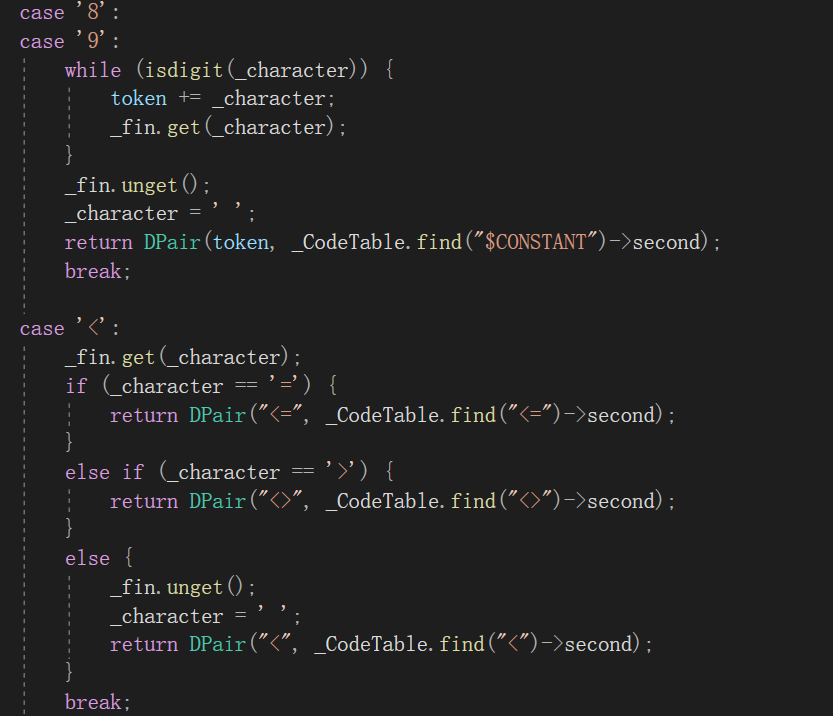
1）对于变量名，开头的一定是字母，接下来可以是字母或数字

2）如果识别出一个标识符全部由字母组成，且不属于任何关键字，则归为变量名，否则归为关键字



#### 分析整型常量

程序到达这里状态是：当前字符一定是数字，且当前字符的前一个字符一定不是数字。

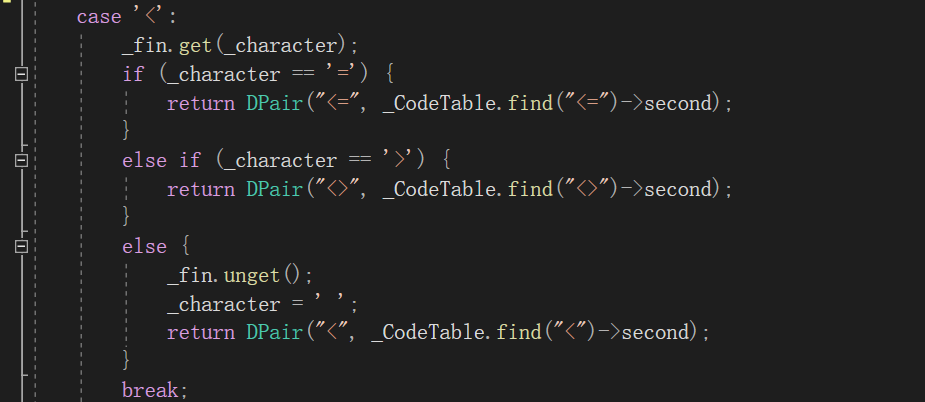
如果要识别完整的整数，需要不断读入字符并保存起来，直到当前字符不是数字为止。

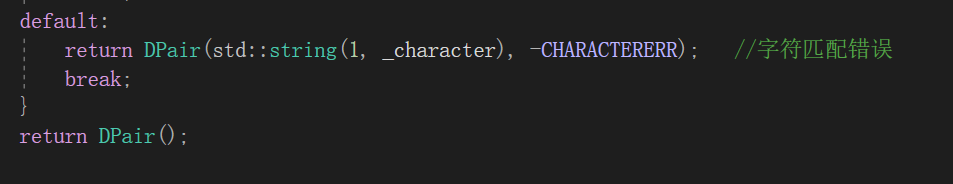
#### 分析操作符

程序到达这里的状态是：当前字符一定是合法操作符的一部分，且当前字符的前一个字符一定不属于任何合法的操作符。

注意：

识别操作符需要最长串匹配原则，如下例读到字符‘<’时，再读入一个字符即可判断是否构成操作符‘<=’, ‘<>’, 如果不匹配，把字符rollback，利用unget()函数实现。



均不匹配，遇到非法字符报错

### 4.3.3报错函数void error(int type)

### 4.3.4输出词法分析结果函数void output(const DPair binseq)

# 5 测试计划

求n！（该测试程序包含算术表达式，逻辑表达式，赋值语句，if-then分支结构，while循环结构）

## 5.1正确测试用例

## 5.2错误测试用例

本测试用例中设计了以下错误：

Begin end不匹配

Interger 标识符错误

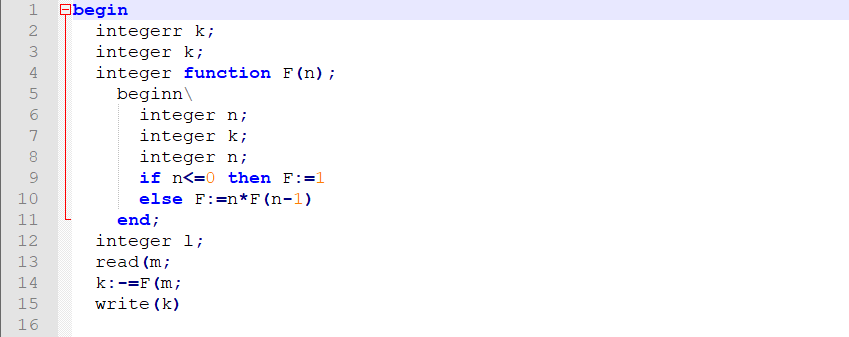
Begin 标识符错误

非法字符\

重复定义变量k, n

未定义变量m

缺少括号）、缺少赋值符号：=

****

## 5.3 实验结果

### 5.3.1词法分析result.dyd输出结果

### 5.3.2错误文件error.err输出