

编译原理

词法分析器实验报告

编译环境：Microsoft Visual Studio 2019

Windows SDK 版本：10.0

平台工具集：Visual Studio 2019 (v142)

语言：C++

需求：词法分析程序的设计和实现。

设计并实现 C 语言的词法分析程序，要求如下：

- (1) 可以识别出用 C 语言编写的源程序中的每个单词符号，并以记号的形式输出每个单词符号。
- (2) 可以识别并读取源程序中的注释。
- (3) 可以统计源程序汇总的语句行数、各类单词的个数、以及字符个数，并输出统计结果。
- (4) 检查源程序中存在的词法错误，并报告错误所在的位置。
- (5) 对源程序中存在的错误进行适当的恢复，使词法分析可以继续进行，对源程序进行一次扫描即可检查并报告源程序中存在的所有词法错误。

核心算法：

读入源程序，将源程序拆分为单词，同时去除空格和空行并存入 vector 中，一个字符一个字符的读入自动机，文件输出单词和类型，并在最后输出统计结果，在屏幕输出错误。

PS:不支持中文字符

源程序的记号和表达式：

1、标识符：字母或“_”开头，后跟字母或数字或“_”组成的符号串

2、关键字：标识符集合的子集。C语言有32个关键字，分别为：

```
{ "do", "double", "auto", "break", "case", "char", "const", "continue", "default", "else", "enum", "extern", "float", "for", "goto", "if", "int", "long", "register", "return", "struct", "switch", "typedef", "short", "signed", "sizeof", "static", "union", "unsigned", "void", "volatile", "while" }。
```

3、数字：由整数部分、可选小数部分和可选整数部分构成

4、运算符：+ * . () ~ & ^ % <= << < >= >> > == /= !&& & || | ->

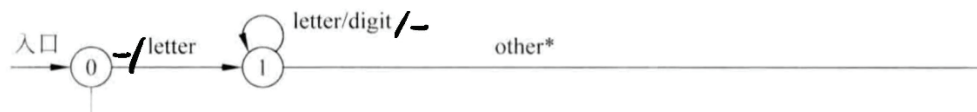
- : ?

5、分隔符：; # ' " [] {},

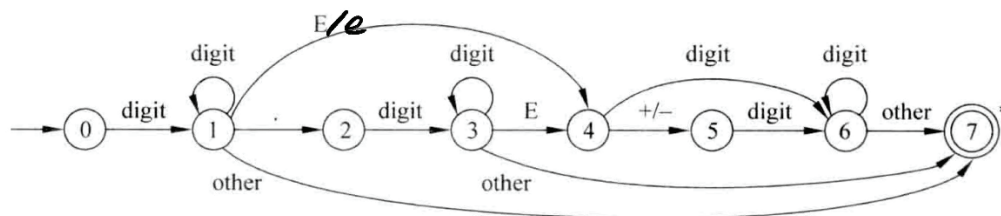
6、转义符：\

状态转换图：

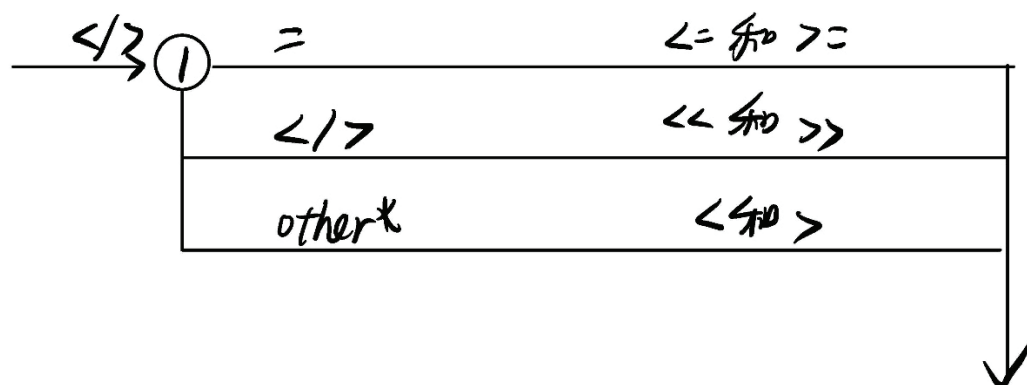
标识符/关键字



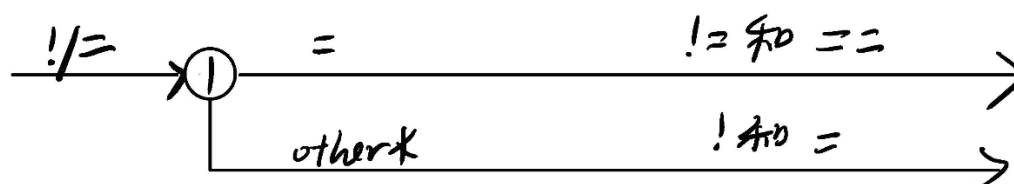
数字



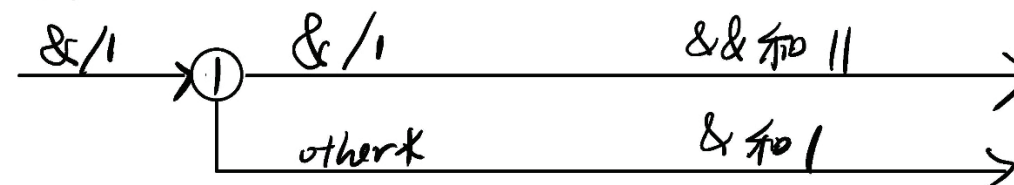
>和<



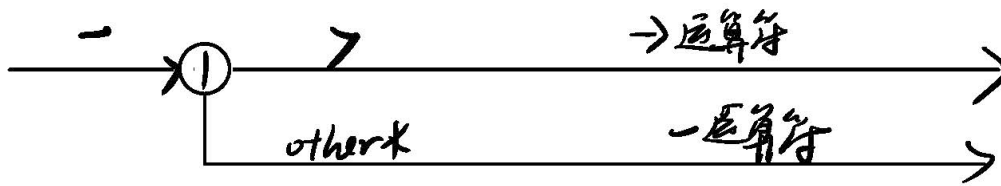
=和!



&和|



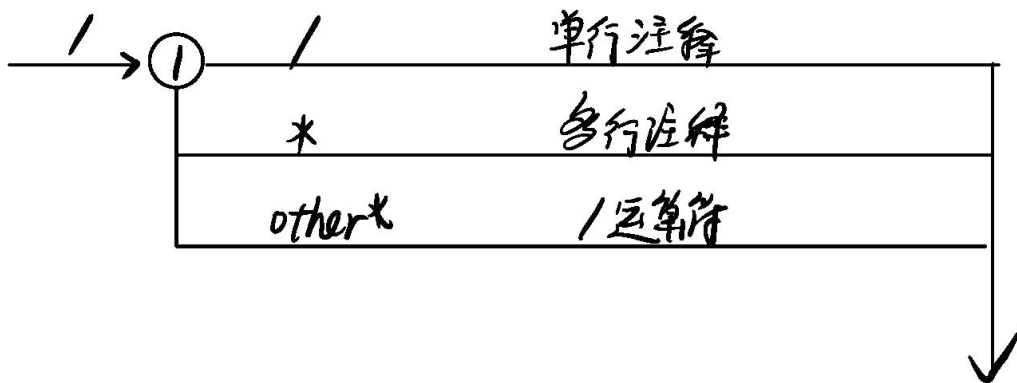
-状态



其他

~ / + / - / * / . / (/ > / ; / # / \ / ' / " / [/] / { / }
/ ^ / % / ? / ! / : / 非法字符 .

注释



输入形式:

文件输入。

输出形式:

分两部分：文件输出和屏幕输出。文件输出分析内容，分四部分：按文件顺序输出类型和表达式，每种表达式数量统计，每种类型数量统计，行数、单词数、字符数统计；屏幕输出错误。详见输出样例部分。

变量和函数:

在附件main.h

```
int start = 0, over = 0, first = 0; // 指针
```

```
int letters = 0, lines = 0, words = 0; // 统计单词数，行数和字符数的变量
```

```
int state; // 状态变量
```

```
class str // 单词类
```

```
{
```

```

public:
    int linenum;//行号
    string Str;//单词
};

//输出文件名
string file_name;

//返回单词或符号,从位置i开始查找,引用参数j返回这个单词最后一个字符在str的位置。
string GetWord(string str, int i, int& j);

//除去字符串中连续的空格和换行。第一个参数为目标字符串,第二个参数为开始位置。返回值为
第一个有效字符在字符串的位置
int get_nbc(string str, int i);

//文件输出函数,成功输出返回true,失败返回false
bool Output(vector<pair<string, string> > v);

//词法分析主要算法函数,返回一个pair型数组
vector<pair<string, string> > analyse(vector<str>& vec);

//此函数判断str是否为关键字,是的话,返回真,反之返回假
bool IsKey(string str);

//此函数判断C是否为字母,是的话,返回真,反之返回假
bool letter(char C);

//此函数判断C是否为数字,是的话,返回真,反之返回假
bool digit(char C);

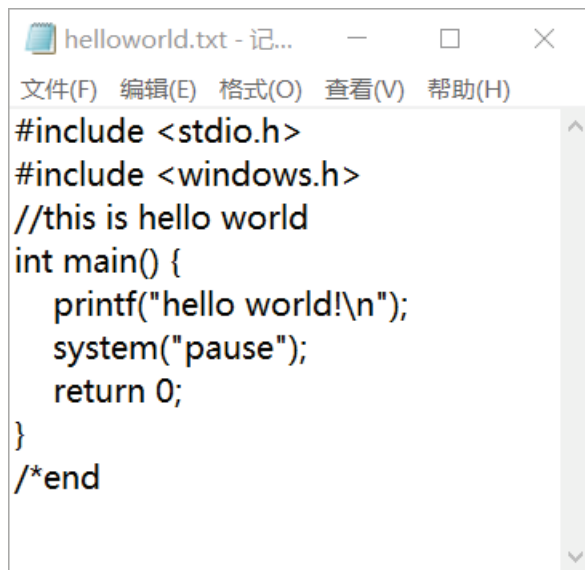
```

详细代码:

详见附件 main.cpp

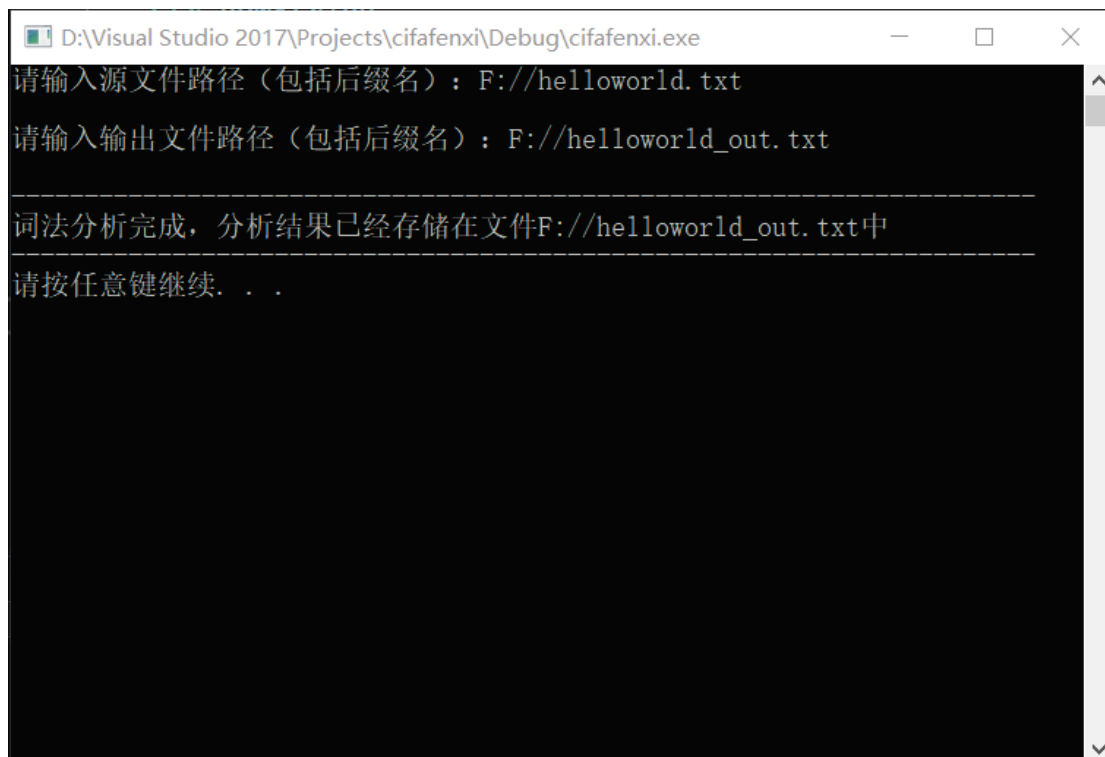
测试样例: 以 helloworld.txt 为例,其余见附件

1、输入



```
#include <stdio.h>
#include <windows.h>
//this is hello world
int main() {
    printf("hello world!\n");
    system("pause");
    return 0;
}
/*end
```

2、输出：



```
D:\Visual Studio 2017\Projects\cifafenxi\Debug\cifafenxi.exe
请输入源文件路径（包括后缀名）： F://helloworld.txt
请输入输出文件路径（包括后缀名）： F://helloworld_out.txt

-----
词法分析完成，分析结果已经存储在文件F://helloworld_out.txt中
-----
请按任意键继续. . .
```

helloworld_out.txt - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

类型	表达式	标识符	pause
分隔符	#	分隔符	"
标识符	include	运算符)
运算符	<	分隔符	;
标识符	stdio	关键字	return
运算符	.	数字	0
标识符	h	分隔符	;
运算符	>	分隔符	}
分隔符	#	表达式	个数
标识符	include	#	2
运算符	<	include	2
标识符	windows	<	2
运算符	.	stdio	1
标识符	h	.	2
运算符	>	h	2
关键字	int	>	2
标识符	main	windows	1
运算符	(int	1
运算符)	main	1
分隔符	{	(3
标识符	printf)	3
运算符	({	1
分隔符	"	printf	1
标识符	hello	"	4
标识符	world	hello	1
运算符	!	world	1
转义符	\	!	1
标识符	n	\	1
分隔符	"	n	1
运算符)	;	3
分隔符	;	system	1
标识符	system	pause	1
运算符	(return	1
分隔符	"	0	1
标识符	pause	}	1
分隔符	"		

类型	个数
分隔符	11
标识符	13
运算符	13
关键字	2
转义符	1
数字	1

源程序共有9行。

源程序共有41个单词。

源程序共有95个字符。

3、错误输出

```
D:\Visual Studio 2017\Projects\cifafenxi\Debug\cifafenxi.exe
请输入源文件路径（包括后缀名）：F://error.txt
第2行有词法错误：1.
第3行有词法错误：1.33ea
第6行有非法字符：@
请输入输出文件路径（包括后缀名）：F://error_out.txt

-----
词法分析完成，分析结果已经存储在文件F://error_out.txt中
-----
请按任意键继续. . .
```