实验一 词法分析

一、实验目的

通过设计编制调试 C 语言的词法分析程序，加深对词法分析原理的理解。并掌握在对程序设计语言源程序进行扫描过程中将其分解为各类单词的词法分析方法。

编制一个读单词过程，从输入的源程序中，识别出各个具有独立意义的单词，即基本字、标识符、常数、运算符、分隔符五大类。并依次输出各个单词的内部编码及单词符号自身值。

C 语言定义了 32 个关键字、34 种运算符，列表如下:

表 1：关键字表

类型	关键字	说明
数据类型修饰相关	auto	按照自动的方式进行变量的存储
	const	定义常量或常参数
	extern	声明外部变量或函数
	register	指定变量的存储类型是寄存器变量
	static	指定变量的存储类型是静态变量，或指定函数是静态函数
	volatile	变量的值可能在程序的外部被改变
数据类型相关	char	字符型数据
	short	定义短整型变量或指针
	int	整型数据
	long	长整型数据
	signed	有符号的整型数据
	unsigned	定义无符号的整型变量或数据
	float	单精度浮点型数据
	double	双精度浮点型数据

	struct	结构体型数据
	enum	枚举型类型
	union	联合型数据
	void	空型数据
	typedef	为数据类型定义别名
流程控制相关	continue	结束本次循环进入下一次循环
	break	跳出循环或 switch 语句
	switch	定义 switch 语句
	case	定义 switch 中的 case 子句
	default	定义 switch 中的 default 子句
	do	定义 do-while 语句
	while	定义 while 或 do-while 语句
	if	定义 if 语句或 if-else 语句
	else	定义 if-else 语句
	for	定义 for 循环语句
	goto	定义 goto 语句
预处理相关	#include	包含头文件
	#define	定义宏
	#undef	取消已经定义的宏
	#if	定义条件编译的条件
	#ifdef	定义条件编译的条件
	#ifndef	定义条件编译的条件
	#elif	定义条件编译的条件
	#endif	结束条件编译

其他	return	从函数返回

表 2: 运算符表

优先级	运算符	含义	操作数数目	结合方向感
1	() [] -> .	括号（函数等），数组，两种结构成员访问	双目	左-右
2	! ~ ++ -- + - |* & (类型) sizeof	否定，按位取反，自增，自减，正负号，间接，取址，类型转换，求大小	单目	右-左
3	* / %	乘，除，取模	双目	左-右
4	+ -	加，减	双目	左-右
5	<< >>	左移，右移	双目	左-右
6	< <= >= >	小于，小于等于，大于等于，大于	双目	左-右
7	== !=	等于，不等于	双目	左-右
8	&	按位与	双目	左-右
9	^	按位异或	双目	左-右
10	|	按位或	双目	左-右
11	&&	逻辑与	双目	左-右
12	||	逻辑或	双目	左-右
13	? :	条件	三目	右-左
14	= += -= *= /= &= ^= |= <<= >>=	各种赋值	双目	右-左
15	,	逗号（顺序）	双目	左-右

实验中，可以取 C 语言关键字和标识符的子集进行词法分析。且称之为 TEST 语言。

二、实验预习提示

1．词法分析器的功能和输出格式

词法分析器的功能是输入源程序，输出单词符号。词法分析器的单词符号常常表示成以下的二元式(单词种别码，单词符号的属性值)。

2．TEST 语言的词法规则

<ID>→<letter>|ID<letter>|ID<digit>

<NUM>→<digit>|NUM <digit>

<letter>→ a|b|…|z|A|B|…|Z

<digit>→1|2|…|9|0

<singleword>→+|-|*|/|=| (|) |{|}|: |, |; |<|>|!

<doubleword>→>=|<=|! =|==

<comment_first>→/*

<comment_last>→*/

<single_comment>→//

三、实验过程和指导

1. 阅读课本有关章节，明确语言的语法，画出状态图和词法分析算法流程图。

2. 编制好程序。

3. 准备好多组测试数据。

4. 程序要求

程序输入/输出示例:

输入如下一段:

```
{
int a, b ;
a = 10 ;
b = a * 6 ;
```

```
}
```

要求输出为:

```
{    {
int  int
ID   a
,    ,
ID   b
;    ;
ID   a
=    =
NUM    10
;    ;
ID   b
=    =
ID   a
*    *
NUM    6
;    ;
}    }
```

5．修改状态图、算法流程图及程序，使得该程序能够识别注释结束符“*/”。

6. 撰写实验报告，报告的文档命名格式 “班级-学号-姓名-词法分析器”

参考程序如下，完成程序，调试运行，给出调试示例及运行结果截图

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
//#include<iostream.h>//
using namespace std;
string keywords[20] = { "include","void","main","int",
                        "char","float","double","if",
                        "else","then","break","continue",
                        "for","do","while","printf","scanf",
                        "begin","end","return" };
char rz[99999] = " ";
string id[10000];
int pp = 0;
string nu[10000];
int qq = 0;

int isLetter(char a)   //判断是否是字母
{
    请完成此函数
}

int isDigit(char a)   //判断是否是数字
{
    请完成此函数
}

int alpha(int st)   //识别保留字和标识符，给此函数的的语句加上注释
{
    char wordbuf[20] = " ";
    int n = 0;
    for (;;)
    {
        wordbuf[n] = rz[st];
        st++;
        n++;
        if ((isDigit(rz[st]) == 1) || (isLetter(rz[st]) == 1) || (rz[st] == '_'))
            wordbuf[n] = rz[st];
        else
            break;
```

```
        }
        int flag = 0;
        for (int k = 0; k < 20; k++)
        {
            if (strcmp(keywords[k].c_str(),wordbuf) == 0)
                flag = 1;
        }
        if (flag == 0)
        {
            int flagg = -1;
            for (int t = 0; t < pp; t++)
            {
                if (strcmp(id[t].c_str(), wordbuf) == 0)
                {
                    flagg = t;
                }
            }
            if (flagg != -1)
                printf(" (id,%d) ", flagg);
            else
            {
                id[pp] = wordbuf;
                printf(" (id,%d) ", pp);
                pp++;
            }
        }
        else
        {
            printf(" (");
            for (int i = 0; i < n; i++)
            {
                printf("%c", wordbuf[i]);
            }
            printf(",-) ");
        }
        return st;
}
int number(int st)   //识别整数
{
        char numbuf[20] = " ";
        int n = 0;
        int k = 0;
        int flag = 0;
        for (;;)
```

```
{
	numbuf[n] = rz[st];
	st++;
	n++;
	if (isDigit(rz[st]) == 1)
	{
		numbuf[n] = rz[st];
	}
	else if ((k == 0) && (rz[st] == '.'))
	{
		numbuf[n] = rz[st];
		k++;
	}
	else if (isLetter(rz[st]) == 1)
	{
		numbuf[n] = rz[st];
		flag = 1;
		continue;
	}
	else
		break;
}
if (flag == 0)
{
	int flagg = -1;
	for (int t = 0; t < qq; t++)
		if (strcmp(nu[t].c_str(), numbuf) == 0)
			flagg = t;
	if (flagg != -1)
		printf(" (nu,%d) ", flagg);
	else
	{
		nu[qq] = numbuf;
		printf(" (nu,%d) ", qq);
		qq++;
	}
}
else {
	printf(" (");
	for (int i = 0; i < n; i++)
		printf("%c", numbuf[i]);
	printf(",error digital!) ");
}
return st;
```

```
}
int anotation(int st)  //处理除号/和注释，给此函数的语句加上注释
{
	char tabuf[9999] = " ";
	int   n = 0;
	st++;
	if (rz[st] == '/')
	{
		printf("    (//,-)");
		st++;
		while (rz[st] != 10)
		{
			tabuf[n] = rz[st];
			st++;
			n++;
		}
		printf(" \n  注释");
		for (int i = 0; i < n; i++)
			printf("%c", tabuf[i]);
	}
	else if (rz[st] == '*')
	{
		printf("    (/*,-)    ");
		st++;
		int stt = st + 1;
		while (1)
		{
			if (rz[st] == '*' && rz[st + 1] == '/')
				break;
			tabuf[n] = rz[st];
			st++;
			n++;
			if (rz[st + 1] == '\0')
			{
				printf("(/* error!!\n)");
				return st + 1;
			}
		}
		printf(" \n  注释");
		for (int i = 0; i < n; i++)
			printf("%c", tabuf[i]);
		printf("    (*/,-)    ");
		st = st + 2;
```

```
        }
        else if (rz[st] == '=')
        {
                st++;
                printf("    (/*,-)    ");
        }
        else printf("    (/,-)     ");
        return st;
}
int other(int st)    //函数识别其他特殊字符
{
        switch (rz[st])
        {
        case'=':
                st++;
                if (rz[st] == '=')
                {
                        st++;
                        printf(" (rlop,==)    ");
                }
                else
                        printf(" (rlop,=)    ");
                break;
        case'+':
                st++;
                if (rz[st] == '=')
                {
                        st++;
                        printf("    (+=,-)    ");
                }
                else if (rz[st] == '+')
                {
                        st++;
                        printf("    (++,-)    ");
                }
                else printf("    (+,-)    ");
                break;
        case'-':
                st++;
                if (rz[st] == '=')
                {
                        st++;
                        printf("    (-=,-)    ");
                }
```

```
        else if (rz[st] == '-')
        {
                st++;
                printf("    (--,-)    ");
        }
        else
                printf("    (-,-)    ");
        break;
case'*':
        st++;
        if (rz[st] == '=')
        {
                st++;
                printf("    (*=,-)    ");
        }

        else
                printf("    (*,-) ");
        break;
case'>':
        st++;
        if (rz[st] == '=')
        {
                st++;
                printf(" (rlop,>=)    ");
        }
        else printf(" (rlop,>)    ");
        break;
case'<':
        st++;
        if (rz[st] == '=')
        {
                st++;
                printf(" (rlop,<=)    ");
        }
        else
                printf(" (rlop,<)    ");
        break;
case'%':
        st++;
        if (rz[st] == '=')
        {
                st++;
                printf("    (\%=,-) ");
```

```
        }
        else
            printf("    (\%,-)    ");
        break;
    case'!':
        st++;
        if (rz[st] == '=')
        {
            st++;
            printf("    (!=,-)    ");
        }
        else
            printf("    (!,wrong thing!)    ");
        break;
    case'&':
        st++;
        if (rz[st] == '&')
        {
            st++;
            printf(" (&&,-)    ");
        }
        else printf(" (&,worng word!)    ");
        break;
    case'|':
        st++;
        if (rz[st] == '|')
        {
            st++;
            printf("    (||,-)    ");
        }
        else
            printf("    ( |,worng word ! )    ");
        break;
    case'{':
        st++;
        printf("    ({,-)    ");
        break;
    case'}':
        st++;
        printf("    (},-)    ");
        break;
    case'(':
        st++;
        printf("    ((,-)    ");
```

```
        break;
case')':
        st++;
        printf("    (),-)    ");
        break;
case'[':
        st++;
        printf("    ([,-)    ");
        break;
case']':
        st++;
        printf("    (],-)    ");
        break;
case':':
        st++;
        printf("    (:,-)    ");
        break;
case'#':st++;
        printf("    (#,-)    ");
        break;
case';':
        st++;
        printf("    (;,-)    ");
        break;
case'.':
        st++;
        printf("    (.,-)    ");
        break;
case',':
        st++;
        printf("    (,,-)    ");
        break;
case' ':
        st++;
        break;
case '    ':
        st++;
        break;
case 10:
        st++;
        printf("\n");
        break;
case34:
        st++;
```

```
            printf(" (\",-)    ");
            break;
        case39:
            st++;
            printf("    (',-)    ");
            break;
        default:
            printf("    (%c,worngthing) ", rz[st]);
            st++;
        }
        return st;
}
int choice(int st)   //根据读入的单词的第一个字符确定调用不同的单词识别函数
{
            if (isLetter(rz[st]) == 1)
            st = alpha(st);
            else if (isDigit(rz[st]) == 1)
```

请填写语句

```
            else if (rz[st] == '/')
```

请填写语句

```
        else
```

请填写语句

```
        return st;
}
int main()
{
        int i = 0;
        FILE* fp;
        char name[10];
        printf("请输入文件名:\n");
        scanf("%s", &name);
        if ((fp = fopen(name, "r")) == NULL)
        {
            printf("Open error!");
            exit(0);
        }
        char ch = fgetc(fp);
        while (ch != EOF)
        {
            rz[i] = ch;
            i++;
```

```
            ch = fgetc(fp);
        }
        fclose(fp);
        int j = 0;
        while (rz[j] != '\0')
            j = choice(j);
        cout << endl << "    程序中标示符如下  " << endl;
        for (i = 0; i < pp; i++)
            cout << i << "      " << id[i] << endl;
        cout << "  程序中数字如下" << endl;
        for (j = 0; j < qq; j++)
            cout << j << "      " << nu[j] << endl;
        system("pause");
}
```

实验报告电子版要求：

一．实验题目：设计基于 C 语言词法分析器

二．实验目的：通过设计编制调试 C 语言的词法分析程序，加深对词法分析原理的理解。并掌握在对程序设计语言源程序进行扫描过程中将其分解为各类单词的词法分析方法。

编制一个读单词过程，从输入的源程序中，识别出各个具有独立意义的单词，即基本字、标识符、常数、运算符、分隔符五大类。并依次输出各个单词的内部编码及单词符号自身值。

三．词法分析器的算法分析：（可用 N-S 图，流程图）

四．程序代码

五．运行结果及分析

六．实验心得