实验 1：词法分析

1.实验目的

设计、编制并调试一个词法分析程序，加深对词法分析原理的理解。

2.实验要求

2.1 待分析的简单语言词法

（1）关键字：int，float，double，long，if，while，else，end，for，等

（2）运算符和界符：{，}，(，)，+，-，*，/，++，--，>=, <=, |, &, ||, &&等

（3）其他单词是标识符（I）和数字（D），通过以下正规式定义：

I=letter(letter|digit)*
N=digit（digit）*

（4）空格由空白、制表符和换行符组成。空格一般用来分隔标识符、数字、运算符、界符和关键字。

2.2 各种单词符号对应的种别码

单词符号	种别码	单词符号	种别码	单词符号	种别码
digit|digit*	0	--	23	struct	45
letter(letter|digit)*	1	&&	24	union	46
#	2	||	25	if	47
<	3	>=	26	else	48
>	4	<=	27	goto	49
(	5	==	28	switch	50

)	6	!=	29	case	51
[	7	>>	30	do	52
]	8	<<	31	while	53
{	9	:=	32	for	54
}	10	int	33	continue	55
,	11	long	34	break	56
;	12	short	35	return	57
+	13	float	36	default	58
-	14	double	37	typedef	59
*	15	char	38	auto	60
/	16	unsigned	39	register	61
%	17	signed	40	extern	62
!	18	const	41	static	63
\'	19	void	42	sizeof	64
"	20	volatile	43	include	65
=	21	enum	44	letter letter*.h	66
\++	22				

2.3 词法分析程序的功能

输入：所给文法的源程序字符串
输出：二元组（syn，token或sum）构成的序列
其中，syn为单词种别码，token为存放单词本身的字符串，sum为二进制整形常数。

3.词法分析程序的算法思想

3.1 实验结果

测试输入：

```
int main (){
/*sdfjhsdjf
*/
int x,y;
char ch;
int    e=5;
int b=2,c;
int i;
      for(i=0;i<5;i++){
         if(e>b){
               c=e+b;
         }
           else
           {
               b++;
         }
      }
      return 0;
}
```

测试输出：

```
第 1 行读进来的是：int main (){
<33,int>
<1,main>
<5,(>
<6,)>
<9,{>
第 2 行读进来的是：/*sdfjhsdjf
第 3 行读进来的是：*/
第 4 行读进来的是：int x,y;
<33,int>
<1,x>
<11,,>
<1,y>
<12,;>
第 5 行读进来的是：char ch;
<38,char>
<1,ch>
<12,;>
第 6 行读进来的是：int    e=5;
<33,int>
<1,e>
<21,=>
<0,binary:101>
<12,;>
第 7 行读进来的是：int b=2,c;
<33,int>
<1,b>
<21,=>
<0,binary:10>
<11,,>
<1,c>
<12,;>
第 8 行读进来的是：int i;
<33,int>
<1,i>
<12,;>
第 9 行读进来的是：        for(i=0;i<5;i++){
<54,for>
<5,(>
<1,i>
<21,=>
<0,binary:0>
<12,;>
<1,i>
<3,<>
<0,binary:101>
<12,;>
<1,i>
<22,++>
<6,)>
<9,{>
第 10 行读进来的是：              c=e+b;
<1,c>
<21,=>
<1,e>
```

<13,+>
<1,b>
<12,;>
第 11 行读进来的是：　　b++;
<1,b>
<22,++>
<12,;>
第 12 行读进来的是：　　}
<10,}>
第 13 行读进来的是：　　return 0;
<57,return>
<0,binary:0>
<12,;>
第 14 行读进来的是：}
<10,}>

3.2 算法思想及总结

（1）本实验程序功能强大，可以对所有 c 语言基础语法文件进行词法分析，可扩展性强，可以轻松扩展后识别 c++等高级语言，鲁棒性强等特点。

（2）本次实验基本任务是把源程序识别出具有独立意义的单词符号，输出对应的二元组，我在设计单词符号和种别码的时候，直接采用 c 语言保留字和运算符，.h 文件采用字符串闭包表示，经过测试，对任意 c 语言文件都可以进行词法分析。

（3）程序设计思路为：首先按行读入字符串后，先对字符串中多余空格进行处理，主要目的是为了减少后续不必要的判断，加快运行速度。处理完空格后对代码中的单行注释进行处理，由于是按行读入，所以多行注释暂时不处理，经过初始化处理掉冗余元素后进行字符串分割，在此处也增加了对多行注释的特判，对于分割后的字符串对我设计的种别码进行 hash 映射输出二元组即可。

（4）本实验中，linux 不支持 gets()函数，可以用 fgets()或者从 c++中 getline()函数替代，scanf()函数不支持输出 string 类型变量可以采用 str.c_str()将 string 类型变量转换为一个临时的 char 常量输出。

4.词法分析程序的 C 语言框架

```
#include<bits/stdc++.h>
```

```
using namespace std;
/*
easy lexical analyzer of zwx
*/
bool isNum(char chr){//判断数字
    return    chr>='0'&&chr<='9';
}

bool isLetter(char chr){//判断字符
    //由于变量名中可以出现下划线，所以这里对下划线也返回 true,对于引入的文件，可能存在.，也加入判断
    return (chr>='a'&&chr<='z')||(chr>='A'&&chr<='Z')||(chr=='_')||(chr=='.');
}
//定界符  单个
char symbol[]={'#','<','>','(',')','[',']','{','}',',',';','+','-','*','/','%','!','\'','"','='};//20 个  0~19 +[2]
string twoSymbol[]={"++","--","&&","||",">=","<=","==","!=",">>","<<",":="};//11 个  0~10+[22]

int isSymbol(char chr){//判断是否是定界符
    for(int i=0;i<20;i++){
        if(chr==symbol[i])return i+2;
    }
    return -1;
}

int isTwoSymbol(string str){//判断是不是两位的符号
    for(int i=0;i<11;i++){
        if(str==twoSymbol[i])return i+22;
    }
    return -1;
}

//关键字  (保留字)reserved word，c 语言中一共有 32 个
string keyword[]={"int","long","short","float","double","char","unsigned","signed","const","void","volatile","enum","struct","union","if","else","goto","switch","case","do","while","for","continue","break","return","default","typedef","auto","register","extern","static","sizeof","include"};
//32 个  0~31+[27]

int isKeyword(string str){//判断关键字
    if(str.size()>2&&str[str.size()-1]=='h'&&str[str.size()-2]=='.')return 66;
    for(int i=0;i<33;i++){
        if(str==keyword[i])return i+33;
    }
    return -1;
```

```
}


const int len=255;
char word[len],temp[len];
string _input;
// 输入字符串 ，空格处理后，注释处理后
void hitSpace()//多余空格处理:只保留间隔用的空格
{
	int tot=0;
	memset(word,0,sizeof(word));
	bool pre=false; //?
	for(int i=0;i<_input.size();i++){
		if(_input[i]==' '&&pre){
			word[tot]=_input[i];
			pre=false;
		}
		else if(_input[i]=='\t'){
				continue;
		}
		else {
			word[tot]=_input[i];
			pre=true;
		}
		tot++;
	}
}
// 行注释处理
void delNotes(){
	memset(temp,0,sizeof(temp));
	for(int i=0;i<strlen(word);i++){
		if(word[i]=='/'&&word[i+1]=='/')break;
		temp[i]=word[i];
	}
}
void dToB(int ans){//十进制
	if(!ans)return ;
	dToB(ans/2);
	printf("%d",ans%2);
}

void init(){
	bool Bnotes=false;//多行注释
	string tempstr;
```

```
    char *str;
    int line=0;//当前行数
    while(getline(cin,_input)){//按行读入，对空格不敏感
        line++;
        hitSpace();//处理多余空格
        delNotes();//处理注释
        printf("第%d 行读进来的是：%s\n",line,temp);
        //printf("%d",line);
         str=strtok(temp," ");//分割字符串
         while(str!=NULL){
            for(int i=0;i<strlen(str);){//遍历此行

                if(*(str+i)=='/'&&*(str+i+1)=='*'){//处理多行注释中的左注释
                        Bnotes=true;
                        break;
                }
                if(*(str+i)=='*'&&*(str+i+1)=='/'&&Bnotes){ //处理多行注释中的右注释
                    Bnotes=false;
                    i+=2;
                    break; //  不应该删除，
                }
                tempstr="";
                if(!Bnotes&&isLetter(*(str+i))){ //出现字符的时候，
                    while(isLetter(*(str+i))||isNum(*(str+i)))//提取字符串
                        tempstr+=*(str+(i++));
                    if(isKeyword(tempstr)==-1)
                        printf("< %d ,%s >\n",1,tempstr.c_str());
                    else if(tempstr[0]=='.')
                        printf(" 第 %d 行 出 现 因 为 小 数 点 作 为 开 头 定 义 变 量 的 错
误:%s\n",line,tempstr.c_str());
                    else
                        printf("< %d ,%s >\n",isKeyword(tempstr),tempstr.c_str());
                    continue;
                }
                if(!Bnotes&&isNum(*(str+i))){
                    while(isNum(*(str+i)))//提取数字串
                        tempstr+=*(str+(i++));
                        int slen=tempstr.size();
                        long long    ans=0;
                        for(int i=0;i<slen;i++){
                        ans=ans*10+(int)(tempstr[i]-'0');//ans 一定在 long long  范围内
                        }
                    printf("< %d ,binary:",0);
                    if(ans)
```

```
                    dToB(ans);
                else
                    printf("0");
                printf(" >\n");//数字的标识符

                continue;
            }
            if(!Bnotes&&isSymbol(*(str+i))!=-1){//定界符和运算符
                if(isSymbol(*(str+i+1))!=-1){
                    string pstr="";
                    pstr+=*(str+i);
                    pstr+=*(str+(i+1));
                    //cout<<pstr<<endl;
                    //printf("now：%s\n",*(str+i)+*(str+i+1));
                    if(isTwoSymbol(pstr)!=-1){
                        printf("< %d ,%s >\n",isTwoSymbol(pstr),pstr.c_str());
                        i++;
                    }
                    else
                        printf("< %d ,%c >\n",isSymbol(*(str+i)),*(str+i));
                }
                else {
                    printf("< %d ,%c >\n",isSymbol(*(str+i)),*(str+i));
                }
                i++;
                continue;
            }
            if(*(str+i)!=' ')
            printf("第%d 行出现错误：%c\n",line,*(str+i));

        }
        str=strtok(NULL," ");
    }
  }
}
int main (){
    freopen("input.cpp","r",stdin);//文件标准读入
    freopen("lexicalResult.txt","w",stdout);//文件标准输出
    init();
    return 0;
}
```