词法分析器

0. 概述

设计，编写并调试一个简单的词法分析程序。

1. 实验设计

1. 对字符串进行分类，定义基本字

```
void map_init(){//对应关系进行初始化
    word["["]="l_square";
    word["]"]="r_square";
    word["("]="l_paren";
    word[")"]="r_paren";
    word["{"]="l_brace";
    word["}"]="r_brace";
    word["."]="period";
    word["..."]="ellipsis";
    word["&"]="amp";
    word["&&"]="ampamp";
    word["&="]="ampequal";
    word["*"]="star";
    word["*="]="starequal";
    word["+"]="plus";
    word["++"]="plusplus";
    word["+="]="plusequal";
    word["-"]="minus";
    word["->"]="arrow";
    word["--"]="minusminus";
    word["-="]="minusequal";
    word["~"]="tilde";
    word["!"]="exclaim";
    word["!="]="exclaimequal";
    word["/"]="slash";
    word["/="]="slashequal";
    word["%"]="percent";
    word["%="]="percentequal";
    word["<"]="less";
    word["<="]="lessequal";
    word[">"]="greater";
    word[">="]="greaterequal";
    word["^"]="caret";
    word["^="]="caretequal";
    word["|"]="pipe";
    word["||"]="pipepipe";
    word["|="]="pipeequal";
    word["?"]="question";
    word[":"]="colon";
    word[";"]="semi";
    word["="]="equal";
    word["=="]="equalequal";
    word[","]="comma";
    word["#"]="hash";
    word["##"]="hashhash";
    word["#@"]="hashat";
    word[".*"]="periodstar";
    word["->*"]="arrowstar";
    word["::"]="coloncolon";
    word["int"]="int";
    word["void"]="void";
    word["if"]="if";
    word["else"]="else";
    word["for"]="for";
    word["return"]="return";
    word["main"]="identifier";
    word["scanf"]="identifier";
}
```

2. 对字符串进行识别，是则输出string_literal

```
        if(str[i]=='"'){
            word1=str[i++];
            while(str[i]!='"'){
                word1+=str[i++];
            }
            word1+=str[i++];
            cout<<"("<<"string_literal"<<","<<word1<<")"<<endl;
            aa[len].id  =  "string_literal";
            aa[len++].s = word1;
        }
```

3. 对标识符和基本字进行识别，若为基本字则输出，否则输出为ident

```
 if(isalpha(str[i])){
            word1=str[i++];
            while(isalpha(str[i])||isdigit(str[i])){
                word1+=str[i++];
            }
            it=word.find(word1);
            if(it!=word.end()){
                cout<<"("<<word[word1]<<","<<word1<<")"<<endl;
                aa[len].id =word[word1];
                aa[len++ ].s = word1;
            }
            else{
                cout<<"(ident"<<","<<word1<<")"<<endl;
                 aa[len].id  =  "ident";
                aa[len++].s = word1;
            }
            i--;
        }
```

4. 判断常数，是则输出number

```
else if(isdigit(str[i])){
```

```
            word1=str[i++];
            while(isdigit(str[i])){
                word1+=str[i++];
            }
            if(isalpha(str[i])){
                cout<<"error!"<<endl;
                break;
            }
            else{
                cout<<"(number"<<","<<word1<<")"<<endl;
                 aa[len].id  = "number";
                aa[len++].s = word1;
            }
            i--;
```

5. 对运算符进行判断

```
}else if(str[i]=='<'){//对<,<=分别进行判断
            word1=str[i++];
            if(str[i]=='>'){
                word1+=str[i];
                cout<<"("<<word[word1]<<","<<word1<<")"<<endl;
                aa[len].id  = word[word1];
                aa[len++].s = word1;
            }else if(str[i]=='='){
                word1+=str[i];
                cout<<"("<<word[word1]<<","<<word1<<")"<<endl;
                aa[len].id  = word[word1];
                aa[len++].s = word1;
            }else if(str[i]!=' '||!isdigit(str[i])||!isalpha(str[i])){
                cout<<"("<<word[word1]<<","<<word1<<")"<<endl;
                aa[len].id  = word[word1];
                aa[len++].s = word1;
            }else{
                cout<<"unknown"<<endl;
                break;
            }
            i--;
        }else if(str[i]=='>'){//对>,>=分别进行判断
            word1=str[i++];
            if(str[i]=='='){
                word1+=str[i];
                cout<<"("<<word[word1]<<","<<word1<<")"<<endl;
                aa[len].id  = word[word1];
                aa[len++].s = word1;
            }else if(str[i]!=' '||!isdigit(str[i])||!isalpha(str[i])){
                cout<<"("<<word[word1]<<","<<word1<<")"<<endl;
                aa[len].id  = word[word1];
                aa[len++].s = word1;
            }else{
                cout<<"unknown"<<endl;
                break;
            }
            i--;
        }else if(str[i]==':'){//对:=进行判断
```

```
            word1=str[i++];
            if(str[i]=='='){
                word1+=str[i];
                cout<<"("<<word[word1]<<","<<word1<<")"<<endl;
                aa[len].id  = word[word1];
                aa[len++].s = word1;
            }else if(str[i]!=' '||!isdigit(str[i])||!isalpha(str[i])){
                cout<<"("<<word[word1]<<","<<word1<<")"<<endl;
                aa[len].id  = word[word1];
                aa[len++].s = word1;
            }else{
                cout<<"unknown"<<endl;
                break;
            }
            i--;
```

6. 对其他字符串进行判断，若不在定义中则输出unknown

```
else{
            word1=str[i];
            it=word.find(word1);
            if(it!=word.end()){
                cout<<"("<<word[word1]<<","<<word1<<")"<<endl;
                aa[len].id  = word[word1];
                aa[len++].s = word1;
            }else{
                cout<<"unknown"<<endl;
                break;
            }
        }
```

2. 运行结果

输入：

```
int main(void){
    int i, n;
    scanf("%d",&n);
    if (n > 10)
        printf("hello world\n");
    else
        printf("no\n");
    for (i = 0; i < 10; ++i){
        n = i + 1;
        n += 2;
    }
    return 0;
}
```

输出：

```
(int,int)
(identifier,main)
(l_paren,()
(void,void)
(r_paren,))
(l_brace,{)
(int,int)
(char_constant,i)
(comma,,)
(char_constant,n)
(semi,;)
(identifier,scanf)
(l_paren,()
(string_literal,"%d")
(comma,,)
(amp,&)
(char_constant,n)
(r_paren,))
(semi,;)
(if,if)
(l_paren,()
(char_constant,n)
(greater,>)
(numeric_constant,10)
(r_paren,))
(char_constant,printf)
(l_paren,()
(string_literal,"hello world\n")
(r_paren,))
(semi,;)
(else,else)
(char_constant,printf)
(l_paren,()
(string_literal,"no\n")
(r_paren,))
(semi,;)
(for,for)
(l_paren,()
(char_constant,i)
(equal,=)
(numeric_constant,0)
(semi,;)
(char_constant,i)
(less,<)
(numeric_constant,10)
(semi,;)
(plus,+)
(plus,+)
(char_constant,i)
(r_paren,))
(l_brace,{)
(char_constant,n)
(equal,=)
(char_constant,i)
(plus,+)
(numeric_constant,1)
(semi,;)
(char_constant,n)
(plus,+)
(equal,=)
(numeric_constant,2)
(semi,;)
(r_brace,})
(return,return)
(numeric_constant,0)
(semi,;)
(r_brace,})
```