词法分析最终报告

1. 处理对象与输入：我所做的语法分析器的处理对象是string类型的一个变量，他存放词法分析的所有token的类型tk\_id，以及token在其对应数组中的位置tk\_num,为此我与词法分析的同学进行了沟通，让他的输出改成我所需要的形式，进而成为我语法分析器的输入

例如：对于测试文件test1.txt：

int main()

{

int a,b,c;

a = b / c

int d,e,f;

d = d - e

if(a<b)

{

d = 1

}

else

{

d = 2

}

while(a==b)

{

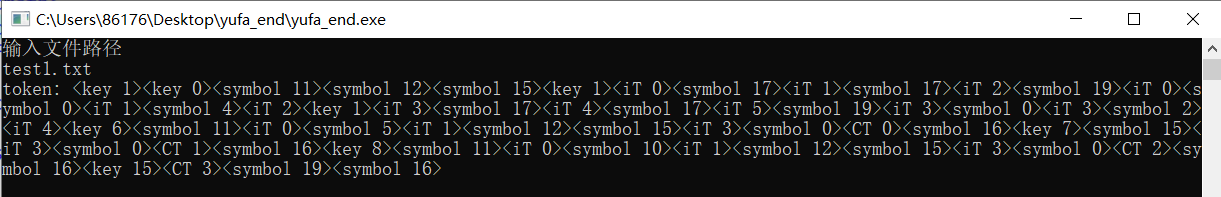
d=3

}

return 0;

}

经过词法分析以后会产生如图所示的这么一个token序列



1. 程序整体功能及输出:

对测试文件所给代码段，根据给定的文法，借助词法分析已经填好的标识符表，常数表等进行递归下降子程序分析，如果测试文件代码段符合文法规则，则输出token已读完，否则会输出对应的错误程序，如果是在主函数program()中的错误，就会输出program wrong后面跟一个数字用于判断具体的出错位置.

例如对于测试文件test2.txt代码段:

int main()

{

int a,b,c;

a = 5

if(a < b)

{

while( c < a)

{

c = a + b

int d;

d = c + 2

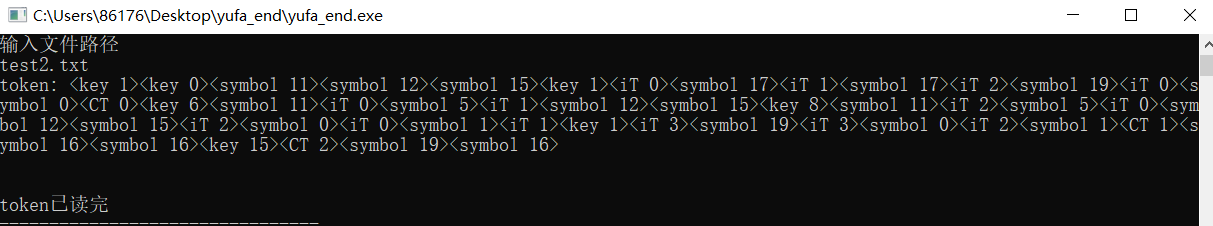
}

}

return 0;

}

会输出如图所示的结果：



可以看到出现了token已读完，表明测试文件符合给定文法的规则，没有出错，语法分析正确。

再看一下测试示例test3.txt:

int main()

{

int a;

int b;

b = 4

int c,d;

c = 10

while(a > b)

{

d = 10

if(c == d)

{

d = a + b + c

}

else

{

d = 5

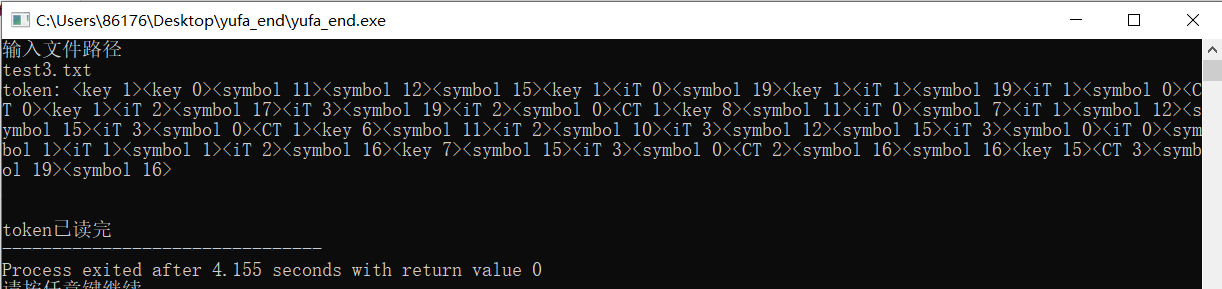
}

}

return 0;

}

输出结果为:



结果正确:

1. 程序源码及相关注释

#include <iostream>

#include <cmath>

#include <cstring>

#include <fstream>

#include "cifa\_end.cpp"

using namespace std;

void item(string& token);

void next\_tk(string& token, string& tk\_id, string& tk\_num, int k)

{

tk\_id.clear();

tk\_num.clear();

int i = 0, j;

//i=1越过<

if (token.size() == 0)

{

cout<<"token已读完";

}

else

{

for (j = 1; j != k; )

{

for (; token[i] != '>'; i++);

i++;

j++;

}

i++;

//当j=k时读第k个token

for (; token[i] != ' '; i++)

{

tk\_id.push\_back(token[i]);

}

for (i++; token[i] != '>'; i++) tk\_num.push\_back(token[i]);

}

}

void erase\_tk(string& token) //删除string中已经分析过的token

{

int i;

int size = token.size();

if (size == 0);

else {

for (i = 0; token[i] != '>'; i++);

token.erase(0, i + 1);

}

}

//<程序> <类型> main() <分程序>

void program(string& token)

{

void type(string& token); //类型

void sub\_program(string& token); //分程序

string tk\_id,tk\_num;

type(token);

next\_tk(token,tk\_id,tk\_num,1);

if(tk\_id == "key" && tk\_num == "0") //main

{

erase\_tk(token);

next\_tk(token,tk\_id,tk\_num,1);

if(tk\_id == "symbol" && tk\_num == "11") //(

{

erase\_tk(token);

next\_tk(token,tk\_id,tk\_num,1);

if(tk\_id == "symbol" && tk\_num == "12") //)

{

erase\_tk(token);

sub\_program(token);

}

else

cout<<"program wrong1";

}

else

cout<<"program wrong2";

}

else

{

cout<<"program wrong3";

cout<<endl;

cout<<token;

}

}

//<类型> int | float | char

void type(string& token)

{

string tk\_id,tk\_num;

next\_tk(token,tk\_id,tk\_num,1);

if(tk\_id == "key" && (tk\_num == "1" || tk\_num == "2" || tk\_num == "3"))

{

erase\_tk(token);

}

else

cout<<"type wrong";

}

//<分程序>->复合语句' 返回

void sub\_program(string& token)

{

void com\_sentence1(string& token); //复合语句'

void back(string& token); //返回

string tk\_id,tk\_num;

next\_tk(token,tk\_id,tk\_num,1);

com\_sentence1(token);

next\_tk(token,tk\_id,tk\_num,1);

back(token);

}

//复合语句' -> {复合语句}|空

void com\_sentence1(string& token)

{

void com\_sentence(string& token);

string tk\_id,tk\_num;

next\_tk(token,tk\_id,tk\_num,1);

while(!(tk\_id == "key" && tk\_num == "15"))

{

com\_sentence(token);

next\_tk(token,tk\_id,tk\_num,1);

}

}

//<复合语句> -> <语句表> | <选择语句> | <循环语句> |空

void com\_sentence(string& token)

{

void sentence\_table(string& token); //语句表

void select\_sentence(string& token); //选择语句

void loop\_sentence(string& token); //循环语句

string tk\_id,tk\_num;

next\_tk(token,tk\_id,tk\_num,1);

if(tk\_id == "symbol" && tk\_num == "15")

{

erase\_tk(token);

}

if(tk\_id == "iT" || (tk\_id == "key" && (tk\_num == "1" || tk\_num == "2" || tk\_num == "3"))) //语句表

{

sentence\_table(token);

}

else if(tk\_id == "key" && (tk\_num == "6" || tk\_num == "7")) //选择语句

{

select\_sentence(token);

}

else if (tk\_id == "key" && tk\_num == "8") //循环语句

{

loop\_sentence(token);

}

}

//<语句表> <赋值语句> ;<语句表> | <变量说明><语句表> | ω

void sentence\_table(string& token)

{

void assign\_sentence(string& token); //赋值语句

void variable(string& token); //变量说明

string tk\_id,tk\_num;

next\_tk(token,tk\_id,tk\_num,1);

if(tk\_id == "iT")

{

assign\_sentence(token);

next\_tk(token,tk\_id,tk\_num,1);

if(tk\_id == "symbol" && tk\_num == "19") //;

{

sentence\_table(token);

}

}

else if(tk\_id == "key" &&(tk\_num == "1" || tk\_num == "2" || tk\_num == "3")) //变量说明

{

variable(token);

}

}

//<赋值语句> <标识符> = <算术表达式>

void assign\_sentence(string& token)

{

void id(string& token); //标识符

void ari\_expression(string& token); //算术表达式

string tk\_id,tk\_num;

next\_tk(token,tk\_id,tk\_num,1);

id(token);

next\_tk(token,tk\_id,tk\_num,1);

if(tk\_id == "symbol" && tk\_num == "0") //=

{

erase\_tk(token);

ari\_expression(token);

}

else

{

cout<<"assign\_sentence wrong3";

}

}

//<标识符>

void id(string& token)

{

string tk\_id,tk\_num;

next\_tk(token,tk\_id,tk\_num,1);

if(tk\_id == "iT")

{

erase\_tk(token);

}

else

cout<<"id wrong";

}

//算术表达式-> <项>{ω0<项>}|ω

void ari\_expression(string& token)

{

string tk\_id,tk\_num;

next\_tk(token,tk\_id,tk\_num,1);

item(token);

next\_tk(token,tk\_id,tk\_num,1);

while(tk\_id == "symbol" &&(tk\_num == "1" || tk\_num == "2")) //+ -

{

erase\_tk(token);

item(token);

next\_tk(token,tk\_id,tk\_num,1);

}

}

//<项> -> <因子>{w1<因子>}

void item(string& token)

{

void factor(string& token);

string tk\_id,tk\_num;

next\_tk(token,tk\_id,tk\_num,1);

factor(token);

next\_tk(token,tk\_id,tk\_num,1);

while (tk\_id == "symbol" && (tk\_num == "3" || tk\_num == "4")) //\* /

{

erase\_tk(token);

item(token);

next\_tk(token,tk\_id,tk\_num,1);

}

}

//因子 -> <标识符>|<常量>|<(算术表达式)>

void factor(string& token)

{

string tk\_id,tk\_num;

next\_tk(token,tk\_id,tk\_num,1);

if(tk\_id == "iT")

{

erase\_tk(token);

}

else if (tk\_id == "CT")

{

erase\_tk(token);

}

else if (tk\_id == "symbol" && tk\_num == "11") // (

{

erase\_tk(token);

ari\_expression(token);

next\_tk(token,tk\_id,tk\_num,1);

if(tk\_id == "symbol" && tk\_num == "12") // )

{

erase\_tk(token);

}

else

cout<<"factor wrong";

}

else

cout<<"factor wrong";

}

//<变量说明> -> <类型><标识符>{,<标识符>};

void variable(string& token)

{

string tk\_id,tk\_num;

next\_tk(token,tk\_id,tk\_num,1);

type(token);

next\_tk(token,tk\_id,tk\_num,1);

id(token);

next\_tk(token,tk\_id,tk\_num,1);

while (tk\_id == "symbol" && tk\_num == "17") // ,

{

erase\_tk(token);

id(token);

next\_tk(token,tk\_id,tk\_num,1);

}

if(tk\_id == "symbol" && tk\_num == "19") // ;

{

erase\_tk(token);

}

else

{

cout<<"variable wrong";

}

}

//<选择语句> -> <选择条件><选择主体>

void select\_sentence(string& token)

{

void select\_condition(string& token); //选择条件

void select\_subject(string& token); //选择主体

string tk\_id,tk\_num;

next\_tk(token,tk\_id,tk\_num,1);

select\_condition(token);

next\_tk(token,tk\_id,tk\_num,1);

select\_subject(token);

}

//<选择条件> -> if (<逻辑表达式>) |else <else分支>

void select\_condition(string& token)

{

void logic\_expression(string& token); //逻辑表达式

void else\_branch(string& token); //else分支

string tk\_id,tk\_num;

next\_tk(token,tk\_id,tk\_num,1);

if(tk\_id == "key" && tk\_num == "6") //if

{

erase\_tk(token);

next\_tk(token,tk\_id,tk\_num,1);

if(tk\_id == "symbol" && tk\_num == "11") //(

{

erase\_tk(token);

logic\_expression(token);

next\_tk(token,tk\_id,tk\_num,1);

if (tk\_id == "symbol" && tk\_num == "12") //)

{

erase\_tk(token);

}

else

cout<<"select\_condition wrong";

}

else

cout<<"select\_condition wrong";

}

else if (tk\_id == "key" && tk\_num == "7") //else

{

erase\_tk(token);

next\_tk(token,tk\_id,tk\_num,1);

else\_branch(token);

}

else

cout<<"select\_condition wrong";

}

//<逻辑表达式> -> <算术表达式>{w4 <算术表达式>} //这里不是循环，是可有可无的意思，不支持多次比较

void logic\_expression(string& token)

{

string tk\_id,tk\_num;

next\_tk(token,tk\_id,tk\_num,1);

ari\_expression(token);

next\_tk(token,tk\_id,tk\_num,1);

if(tk\_id == "symbol" && (tk\_num == "5" || tk\_num == "6" || tk\_num == "7" || tk\_num == "8" || tk\_num == "9" || tk\_num == "10"))// < <= > >= != ==

{

erase\_tk(token);

ari\_expression(token);

}

}

// <else分支> -> if (<逻辑表达式>) | ω

void else\_branch(string& token)

{

string tk\_id,tk\_num;

next\_tk(token,tk\_id,tk\_num,1);

if (tk\_id == "key" && tk\_num == "6") //if

{

erase\_tk(token);

next\_tk(token,tk\_id,tk\_num,1);

if(tk\_id == "symbol" && tk\_num == "11") //(

{

erase\_tk(token);

logic\_expression(token);

next\_tk(token,tk\_id,tk\_num,1);

if (tk\_id == "symbol" && tk\_num == "12") //)

{

erase\_tk(token);

}

else

cout<<"else wrong";

}

else

cout<<"else wrong";

}

}

//<选择主体> -> ; | <复合语句> | {<复合语句>}

void select\_subject(string& token)

{

string tk\_id,tk\_num;

next\_tk(token,tk\_id,tk\_num,1);

if (tk\_id == "symbol" && tk\_num == "19") //;

{

erase\_tk(token);

}

else if ((tk\_id == "iT" || (tk\_id == "key" && (tk\_num == "1" || tk\_num == "2" || tk\_num == "3"))) || (tk\_id == "key" && (tk\_num == "7" || tk\_num == "8")) || (tk\_id == "key" && tk\_num == "9"))

{

erase\_tk(token);

com\_sentence(token);

}

else if(tk\_id == "symbol" && tk\_num == "15") //{

{

erase\_tk(token);

com\_sentence(token);

next\_tk(token,tk\_id,tk\_num,1);

if (tk\_id == "symbol" && tk\_num == "16") //}

{

//////////////////

erase\_tk(token);

}

else

{

cout<<"select\_subject wrong1";

}

}

else

cout<<"select\_subject wrong2";

}

//<循环语句> -> <循环条件> <循环主体>

void loop\_sentence(string& token)

{

void loop\_condition(string& token); //循环条件

void loop\_subject(string& token); //循环主体

string tk\_id,tk\_num;

next\_tk(token,tk\_id,tk\_num,1);

loop\_condition(token);

next\_tk(token,tk\_id,tk\_num,1);

loop\_subject(token);

}

//<循环条件> -> while (<逻辑表达式>)

void loop\_condition(string& token)

{

string tk\_id,tk\_num;

next\_tk(token,tk\_id,tk\_num,1);

if (tk\_id == "key" && tk\_num == "8") //while

{

erase\_tk(token);

next\_tk(token,tk\_id,tk\_num,1);

if(tk\_id == "symbol" && tk\_num == "11") //(

{

erase\_tk(token);

logic\_expression(token);

next\_tk(token,tk\_id,tk\_num,1);

if(tk\_id == "symbol" && tk\_num == "12") //)

{

erase\_tk(token);

}

else

cout<<"loop wrong1";

}

else

cout<<"loop wrong2";

}

else

cout<<"loop wrong3";

}

//<循环主体> -> {{<复合语句>}}

void loop\_subject(string& token)

{

string tk\_id,tk\_num;

next\_tk(token,tk\_id,tk\_num,1);

if(tk\_id == "symbol" && tk\_num == "15") //{

{

erase\_tk(token);

next\_tk(token,tk\_id,tk\_num,1);

while((tk\_id == "iT" || (tk\_id == "key" && (tk\_num == "1" || tk\_num == "2" || tk\_num == "3"))) || (tk\_id == "key" && (tk\_num == "6" || tk\_num == "7")) || (tk\_id == "key" && tk\_num == "8"))

{

com\_sentence(token);

next\_tk(token,tk\_id,tk\_num,1);

}

next\_tk(token,tk\_id,tk\_num,1);

if (tk\_id == "symbol" && tk\_num == "16") //}

{

erase\_tk(token);

}

else

{

cout<<"loop worng4";

cout<<endl;

cout<<token;

}

}

else

cout<<"loop wrong5";

}

//<返回> ->return <常数> | <标识符>;

void back(string& token)

{

string tk\_id,tk\_num;

next\_tk(token,tk\_id,tk\_num,1);

if(tk\_id == "key" && tk\_num == "15") //return

{

erase\_tk(token);

next\_tk(token,tk\_id,tk\_num,1);

if(tk\_id == "CT" || tk\_id == "iT") //常数或标识符

{

erase\_tk(token);

next\_tk(token,tk\_id,tk\_num,1);

if(tk\_id == "symbol" && tk\_num == "19") //;

{

erase\_tk(token);

}

else

cout<<"back wrong1";

}

else

cout<<"back wrong2";

}

else

{

cout<<"back wrong3";

}

}

int main()

{

init\_();

string s,token; //token存放整个token序列

string tk\_id,tk\_num; //token的类型和在数组中的位置

s = read\_file(s);

scanner(s,token); //扫描目标文件，查填表并把token序列送入token string

cout<<"token: "<<token<<endl<<endl<<endl;

program(token);

erase\_tk(token);

next\_tk(token,tk\_id,tk\_num,1);

}