中山大学数据科学与计算机学院本科生实验报告

课程名称：编译器构造实验 任课教师：陈炬桦 教学助理（TA）：

|  |  |  |  |
| --- | --- | --- | --- |
| 学年度 | 2018-2019 | 学期 | 第二学期 |
| 年级 | 2016级 | 专业（方向） | 计算机科学与技术 |
| 学号 | 16337233 | 姓名 | 王凯祺 |
| 电话 | 13434386182 | Email | [Wangkq3@mail2.sysu.edu.cn](mailto:Wangkq3@mail2.sysu.edu.cn) |
| 开始日期 | 2019/4/12 | 完成日期 | 2019/4/12 |

1. 实验题目

**LL(1)语法分析程序**

**1.1 Description**

输入开始符号，非终结符，终结符，产生式，LL(1)分析表  
输出LL(1)分析表

**1.2 Input**

输入开始符号；  
非终结符个数，非终结符，空格符分隔；  
终结符个数，终结符，空格符分隔；  
产生式的个数，各产生式的序号，产生式的左边和右边符号，空格符分隔；  
LL(1)分析表中的产生式个数，序号，行符号，列符号，产生式编号，空格符分隔；  
输入一个算术式符号串，用#结束

**1.3 Output**

输出推导过程，每一步一行，中间“ & ”前是已经识别的子串，后是栈中信息。

2. 算法描述(介绍程序模块功能；流程图)

(1) 根据产生式、LL(1)分析表构造LL(1)分析表。

(2) 最初，分析器把文法的开始符号S置于栈顶（输入串以#结束）。

(3) 若栈顶为终结符，且与当前输入符号匹配，则读头前进一个位置，并逐出栈顶符号，否则报错。

(4) 若栈顶符号是一非终结符U，且当前的输入符号为a，则查看分析表M，若M[U,a]置有关U的产生式U🡪 w，则先从栈中逐出U再把w下推进栈；若w=k，则不推进任何信息进栈，仅逐出栈顶符号；若M[U,a]为空白，则报错。

(5) 重复步骤(3)(4)，直至栈变为空。

(6) 分析器停止于空栈。

3. 测试数据

样例输入

E

6 E A T B F D

9 + - \* / ( ) x y z

13

1 E TA

2 A +TA

3 A -TA

4 A k

5 T FB

6 B \*FB

7 B /FB

8 B k

9 F (E)

10 F D

11 D x

12 D y

13 D z

25

1 E ( 1

2 E x 1

3 E y 1

4 E z 1

5 A + 2

6 A - 3

7 A ) 4

8 A # 4

9 T ( 5

10 T x 5

11 T y 5

12 T z 5

13 B + 8

14 B - 8

15 B \* 6

16 B / 7

17 B ) 8

18 B # 8

19 F ( 9

20 F x 10

21 F y 10

22 F z 10

23 D x 11

24 D y 12

25 D z 13

(x+(y-x\*z)\*(y+x\*z))+x/z#

样例输出

# & E#

# & TA#

# & FBA#

# & (E)BA#

#( & E)BA#

#( & TA)BA#

#( & FBA)BA#

#( & DBA)BA#

#( & xBA)BA#

#(x & BA)BA#

#(x & A)BA#

#(x & +TA)BA#

#(x+ & TA)BA#

#(x+ & FBA)BA#

#(x+ & (E)BA)BA#

#(x+( & E)BA)BA#

#(x+( & TA)BA)BA#

#(x+( & FBA)BA)BA#

#(x+( & DBA)BA)BA#

#(x+( & yBA)BA)BA#

#(x+(y & BA)BA)BA#

#(x+(y & A)BA)BA#

#(x+(y & -TA)BA)BA#

#(x+(y- & TA)BA)BA#

#(x+(y- & FBA)BA)BA#

#(x+(y- & DBA)BA)BA#

#(x+(y- & xBA)BA)BA#

#(x+(y-x & BA)BA)BA#

#(x+(y-x & \*FBA)BA)BA#

#(x+(y-x\* & FBA)BA)BA#

#(x+(y-x\* & DBA)BA)BA#

#(x+(y-x\* & zBA)BA)BA#

#(x+(y-x\*z & BA)BA)BA#

#(x+(y-x\*z & A)BA)BA#

#(x+(y-x\*z & )BA)BA#

#(x+(y-x\*z) & BA)BA#

#(x+(y-x\*z) & \*FBA)BA#

#(x+(y-x\*z)\* & FBA)BA#

#(x+(y-x\*z)\* & (E)BA)BA#

#(x+(y-x\*z)\*( & E)BA)BA#

#(x+(y-x\*z)\*( & TA)BA)BA#

#(x+(y-x\*z)\*( & FBA)BA)BA#

#(x+(y-x\*z)\*( & DBA)BA)BA#

#(x+(y-x\*z)\*( & yBA)BA)BA#

#(x+(y-x\*z)\*(y & BA)BA)BA#

#(x+(y-x\*z)\*(y & A)BA)BA#

#(x+(y-x\*z)\*(y & +TA)BA)BA#

#(x+(y-x\*z)\*(y+ & TA)BA)BA#

#(x+(y-x\*z)\*(y+ & FBA)BA)BA#

#(x+(y-x\*z)\*(y+ & DBA)BA)BA#

#(x+(y-x\*z)\*(y+ & xBA)BA)BA#

#(x+(y-x\*z)\*(y+x & BA)BA)BA#

#(x+(y-x\*z)\*(y+x & \*FBA)BA)BA#

#(x+(y-x\*z)\*(y+x\* & FBA)BA)BA#

#(x+(y-x\*z)\*(y+x\* & DBA)BA)BA#

#(x+(y-x\*z)\*(y+x\* & zBA)BA)BA#

#(x+(y-x\*z)\*(y+x\*z & BA)BA)BA#

#(x+(y-x\*z)\*(y+x\*z & A)BA)BA#

#(x+(y-x\*z)\*(y+x\*z & )BA)BA#

#(x+(y-x\*z)\*(y+x\*z) & BA)BA#

#(x+(y-x\*z)\*(y+x\*z) & A)BA#

#(x+(y-x\*z)\*(y+x\*z) & )BA#

#(x+(y-x\*z)\*(y+x\*z)) & BA#

#(x+(y-x\*z)\*(y+x\*z)) & A#

#(x+(y-x\*z)\*(y+x\*z)) & +TA#

#(x+(y-x\*z)\*(y+x\*z))+ & TA#

#(x+(y-x\*z)\*(y+x\*z))+ & FBA#

#(x+(y-x\*z)\*(y+x\*z))+ & DBA#

#(x+(y-x\*z)\*(y+x\*z))+ & xBA#

#(x+(y-x\*z)\*(y+x\*z))+x & BA#

#(x+(y-x\*z)\*(y+x\*z))+x & /FBA#

#(x+(y-x\*z)\*(y+x\*z))+x/ & FBA#

#(x+(y-x\*z)\*(y+x\*z))+x/ & DBA#

#(x+(y-x\*z)\*(y+x\*z))+x/ & zBA#

#(x+(y-x\*z)\*(y+x\*z))+x/z & BA#

#(x+(y-x\*z)\*(y+x\*z))+x/z & A#

#(x+(y-x\*z)\*(y+x\*z))+x/z & #

4. 程序清单

#include <bits/stdc++.h>

#define N 102

using namespace std;

int n, m, k;

char S, s[N], t[N];

pair<char, string> p[N];

char tmp[N];

int mps[256], mpt[256];

string ans[N][N];

vector<char> stk;

int main() {

scanf("%s", tmp);

S = tmp[0];

scanf("%d", &n);

for (int i = 1; i <= n; ++i) {

scanf("%s", tmp);

s[i] = tmp[0];

mps[s[i]] = i;

}

scanf("%d", &m);

for (int i = 1; i <= m; ++i) {

scanf("%s", tmp);

t[i] = tmp[0];

mpt[t[i]] = i;

}

++ m;

t[m] = '#';

mpt['#'] = m;

scanf("%d", &k);

for (int i = 1; i <= k; ++i) {

int x;

scanf("%d", &x);

scanf("%s", tmp);

scanf("%s", tmp + 1);

p[x] = make\_pair(tmp[0], (string)(tmp + 1));

}

scanf("%d", &k);

for (int i = 1; i <= k; ++i) {

int x;

scanf("%d", &x);

scanf("%s", tmp);

scanf("%s", tmp + 1);

scanf("%d", &x);

ans[ mps[tmp[0]] ][ mpt[tmp[1]] ] = p[x].second;

}

scanf("%s", tmp);

stk.push\_back('#');

stk.push\_back(S);

int len = strlen(tmp);

for (int i = 0; i <= len; ++i) {

int first = 1;

do {

putchar('#');

for (int j = 0; j < i - first; ++j) putchar(tmp[j]);

printf(" & ");

for (int j = (int)stk.size() - 1; j >= 0; --j) putchar(stk[j]);

puts("");

first = 0;

int x = mps[stk.back()];

stk.pop\_back();

int y = mpt[tmp[i]];

for (int j = (int)ans[x][y].size() - 1; j >= 0; --j)

if (ans[x][y][j] != 'k')

stk.push\_back(ans[x][y][j]);

} while (stk[stk.size() - 1] != tmp[i]);

}

}