**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 | 16337327 | 姓名 | 郑映雪 |
| 电话 | 18956073450 | Email | 646352420@qq.com |
| 开始日期 | 2019.3.8 | 完成日期 | 2019.3.8 |

**实验题目**

## 1. Description

输入开始符号，非终结符，终结符，产生式

按非终结符顺序输出产生式；

## 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，各产生式的左边和右边符号，空格符分隔；

## 3. Output

G[开始符号]：

按非终结符顺序输出各产生式；

**算法描述**

## 1. 流程总览

## 2.各模块功能

由以上的流程图可以看出程序由以下几个模块组成：

**a.初始模块**

在进行文法输出之前，需要进行准备工作。首先，对终结符、非终结符和产生式建立表,并进行输入操作。考虑到输出是比较复杂的串集，所以输出集定义为string类型的数组result比较方便。

**b.判重模块**

对于输入的每一个产生式，都有左部和右部，我们想把这些产生式归结起来，就需要根据左部进行分类。一个很方便的办法就是，对于每个输入的产生式的左部，都在result进行判重操作。如果result里已经有该左部的产生式信息（即result[index][0]=PL[0]）,则直接在该项的字符串上进行“加”操作，加上分隔符“|”和右部。如果result里没有该左部，就在result里建立新项，并对于该新项进行左部、等于符号和右部的“加”操作。

**c.输出模块**

题目要求按照非终结符输入的顺序来输出。所以我们在归类完毕之后，还需要按照非终结符输入顺序对result里的符合该非终结符的项进行输出，以达到按非终结符输入顺序重新输出的目的。

**d. 额外模块**

本题没有要求进行非法输入的判断。但是在生活中，非法输入的判断是需要进行的。对于每一个输入的左部和右部，我们要判断是否在非终结符∪终结符列表中。由于题目本身没有要求，所以此处为额外模块的部分，但仍为必要的部分。

**测试数据(2组)**

|  |  |  |
| --- | --- | --- |
| 测试数据 | 测试结果 | 结果判定 |
|  |  | 正确 |
|  |  | 正确 |

**程序清单**

1. #include <iostream>
2. #include <string>
3. **using** **namespace** std;
4. **struct** { **int** Nv; string VN[10]; } Vns = { 0};
5. **struct** { **int** Nt; string VT[10]; } Vts = { 0};
6. **struct** { **int** Np; string PL[20], PR[20]; } ps = { 0 };
7. string S;
8. **int** judge(string left, string right, string result[], **int** len) {
9. **for** (**int** j = 0; j < len; ++j) {
10. **if** (left[0] == result[j][0]) **return** j;
11. }
12. **return** -1;
13. }
14. **int** main(){
15. cin >> S>>Vns.Nv;
16. **for** (**int** i = 0; i < Vns.Nv; ++i)
17. cin >> Vns.VN[i];
18. cin >> Vts.Nt;
19. **for** (**int** i = 0; i < Vts.Nt; ++i)
20. cin >> Vts.VT[i];
22. cin >> ps.Np;
23. string result[20];
24. **int** len=0;
25. **for** (**int** i = 0; i < ps.Np; ++i) {
26. cin >> ps.PL[i] >> ps.PR[i];
27. **int** index = judge(ps.PL[i], ps.PR[i], result, len);
28. **if** (index!=-1) {
29. result[index] += " | ";
30. result[index] += ps.PR[i];
31. }
32. **else** {
33. result[len] = ps.PL[i] + "::=" + ps.PR[i];
34. len++;
35. }
36. }
37. cout << "G[" << S << "]:" << endl;
38. **for** (**int** j = 0; j < Vns.Nv; ++j) {
39. **for** (**int** i = 0; i < len; ++i) {
40. **if** (Vns.VN[j][0] == result[i][0])
41. cout << result[i] << endl;
42. }
43. }
44. }