《编译原理》实验

语法分析实验报告

**学 号：1320151098**

**班 级：07121502**

**姓 名：赵璐**

1. **实验内容及目的**

语法分析是编译程序的核心部分。语法分析的任务是：按照语言的语法规则，对单词串形式的源程序进行语法检查，并识别出相应的语法成分。

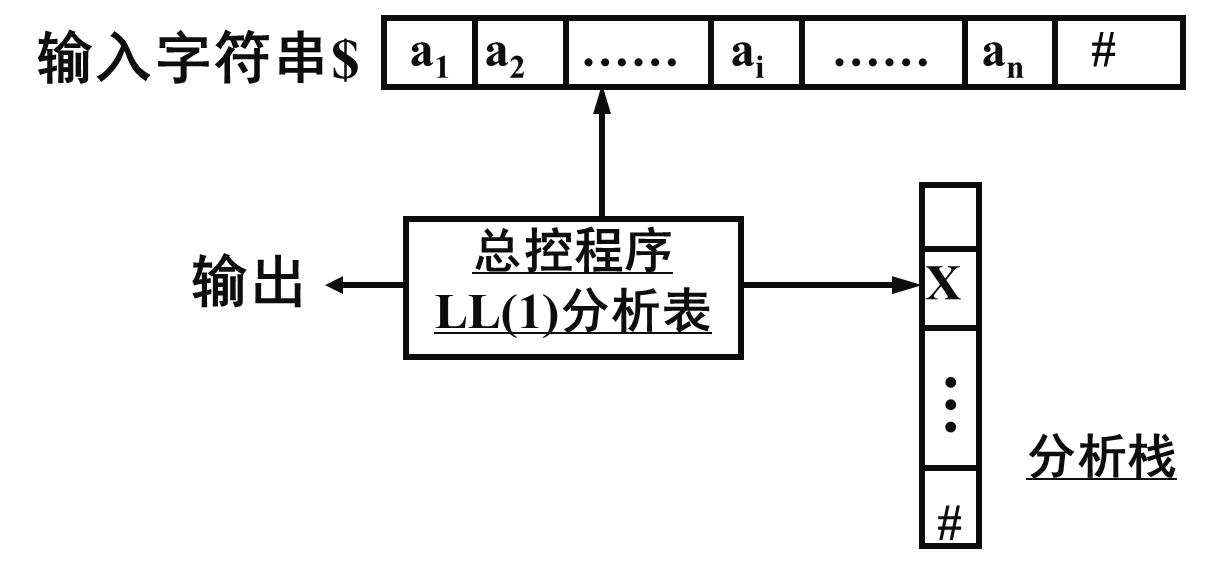
本次实验的目的，是要设计一个简单的C语言的语法分析程序模型，需要完成的功能为：依据C语言基本语法，处理词法分析器的输出，即属性字流形式的源程序，最终识别出无语法错误的语法成分。

本次实验的目的，通过实验掌握语法分析的方法并理解语法分析在整个编译程序中的地位和重要性。

1. **实验方法**

LL(1)分析法是一种自上而下分析法，使用显式栈而不是递归调用来实现分析，判断读入的字符串是否属于该文法的某一句子。

LL(1)分析法的实现是由一个总控程序控制输入字符串在一张LL(1)分析表和一个分析栈上运行而完成语法分析任务的。所以，一个LL(1)分析器的逻辑结构如下图所示，由总控程序、LL(1)分析表和分析栈组成。



1. **实验过程**
2. **文法定义**

文法定义是该实验的最基本步骤，文法定义的简单正确会为后续的编程奠定良好的基础。

C语言具有非常复杂的文法定义，有的文法的产生式很多，需要编译器根据读取的字符串进行移进和规约。我们这里仅仅对C语言中的部分文法进行代码实现。



1. **实验基本思路**
2. 构建二维数组，即LL(1)分析表
3. 初始化用来表示分析栈的一维数组，依次将“#”和文法开始符号压入栈
4. 读取经过词法分析得到的字符串作为输入串，以“#”结束
5. 取出分析栈的栈顶字符X，输入串的第一个字符a

1）X为终结符：

①X=a=“#”，表示分析成功，停止分析过程；

②X=a≠“#”，将X从分析栈顶退掉，指向输入字符串的指针下移一个字符；

③X≠a，出错。

2）X为非终结字符，查分析表，对M(X,a)：

①若M(X,a)中为一个产生式规则，则将X从栈中弹出，并将此规则右部的符号序列推进栈；

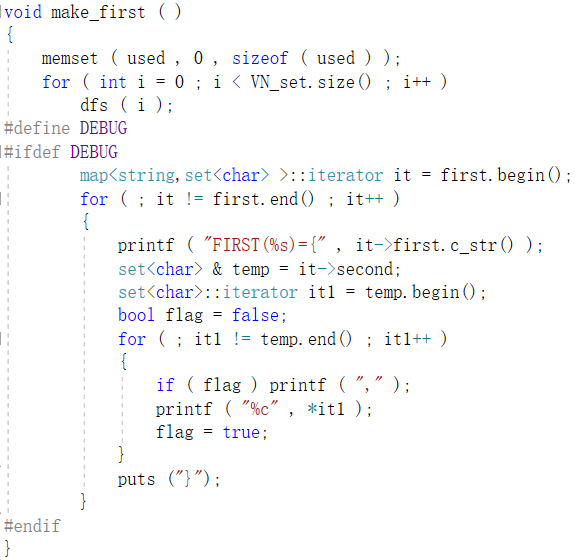
②若M(X,a)中为空白，表示出错。

1. **实验部分实现及代码**
2. **First集**

**算法:**

* 若X->a，则将终结符ａ加入FIRST(X)中;
* 若X->e,则将终结符e加入FIRST(X)中(e表示空串)；
* 若 X->BC…E,则将First(B)所有元素（除了空串）加入 First(X),然后检测First(B),若First(B)中不存在空集,即e,则停止。若存在则向B的后面查看。将First(C)中所有元素（除了空串）加入First(X),然后再检测First(C)中是否有e，重复上述。若E之前的所有非终结符的First集中都含有e,则检测到E时，将First(E)也加入First（X）,若First(E)中含有e,则将 e加入First(X）。

**代码实现：**



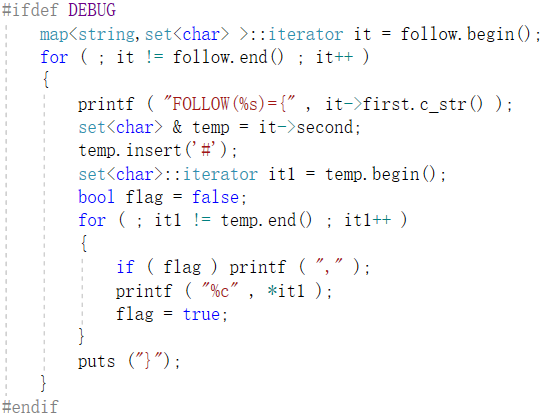
但是若FIRST集中包含空串，此时需要用到FOLLOW集，构造如下。

1. **Follow集**

**算法：**

* 对文法开始符号S,置#于FOLLOW(S)中；
* 对于产生式：A->aBC,将除去空集e的First(C)加入Follow(B)中;
* 对于产生式：A->aB或者A->aBC,(其中C可以推导出空串，C=>\*e),则将Follow(A)加入Follow(B)中。

**代码实现：**



1. **分析栈以及生成树的构造**

在构造生成树时，采用构造前序遍历的N叉树的形式产生，通过设计Tree类和结点CTreeNode、CLeafNode结构构造树，相关调用函数为Tree：：系列函数，具体过程见源代码。

1. **xml文件的生成**

根据构造的N叉树，我们利用每个子节点与父节点之间的关系，采用前序遍历优先搜索，根据每个结点的属性构造相应的xml文件层次，相关代码见源码。

1. **实验结果及分析**
2. **测试代码**
3. **运行结果**
4. **心得体会**

起初选择了LR(0)分析法，但由于其求闭包、构造项目集规范族等步骤难度较大，我不得不放弃。退而求其次，最终选择了LL(1)分析法，虽然对于语法分析器来讲该算法存在一定的漏洞，但也算基本实现的语法分析器的主要功能。怎么讲呢，深刻体会到了力不从心的感觉，编程功底还需要继续加强。

感触最深的是，在设计C语言文法的时候，看惯了书中的“S—>a|B”等，对英文的C语言文法理解起来有些困难，并且LL(1)文法要求其LL(1)分析表不含有多重定义入口，加大了文法设计的难度。但通过查阅资料，求助同学还是凭着自己的理解设计出了符合本实验要求的文法。为后续的编程奠定了良好的基础。

通过动手实践，用一行行代码实现书本上的文字算法，使我对自上而下分析方法的基本概念有了更加准确地认识。由于之前设计大型程序的经验比较少，导致程序结构不够清晰，代码不够模块化，使得调试改错时十分困难，容易牵一发而动全身。并且在实现的过程中，深刻体会到了在为不同的数据选择相应的数据结构时，应该选择实现简单，使用方便的结构，以达到简化代码的目的。总之，在规定时间内代码虽然基本完成，但总是存在一些自己不满意的地方。

下一次设计代码生成时，我会汲取经验教训，在开始敲代码之前，将程序的整体框架构思好，所谓磨刀不误砍柴工。