

**LL（1）分析器**

**课 程： 编译原理**

**姓 名： 侯佳耀**

**学 号： 20182131055**

**班 级： 计科4班**

**设计时间：2021年11月15日**

|  |
| --- |
| **一．实验内容及目的**  一、实验内容：  设计一个应用软件，以实现文法的化简及各种问题的处理。  二、实验要求：  （1）系统需要提供一个文法编辑界面，让用户输入文法规则（可保存、打开存有文法规则的文件）  （2）化简文法：检查文法是否存在有害规则和多余规则并将其去除。系统应该提供窗口以便用户可以查看文法化简后的结果。  （3）检查该文法是否存在着左公共因子（可能包含直接和间接的情况）。如果存在，则消除该文法的左公共因子。系统应该提供窗口以便用户可以查看消除左公共因子的结果。  （4）检查该文法是否存在着左递归（可能包含直接和间接的情况），如果存在，则消除该文法的左递归。系统应该提供窗口以便用户可以查看消除左递归后的结果。  （5）求出经过前面步骤处理好的文法各非终结符号的first集合与follow集合，并提供窗口以便用户可以查看这些集合结果。  （6）对输入的句子进行最左推导分析，系统应该提供界面让用户可以输入要分析的句子以及方便用户查看最左推导的每一步推导结果。  （7）应该书写完善的软件文档  **注：**文法规则为了处理上的简单，输入时可以只输入单一个大写字母作为非终结符号，单一个小写字母作为终结符号，用埃普西隆符号表示空串。如：E->E+T T->a |
| **二．实验文档：**  1.设计思路  首先通过扫描文件的方式输入文法规则，通过过滤方法将相关的各个集合分出。然后根据这些数据进行数据处理，生成无有害，无左递归，无左公因子的文法再次输出。并求改写后的文法的first集合和follow集合。最后根据LL（1）分析表的填表算法，将相关数据填入LL（1）分析表中。最后再通过龙书上的分析栈算法，实现分析是否可由文法最左推导得到某子串的功能。   1. 实验过程   2.1数据结构：  数据结构设计如图    关于其中子串的解释可见文件夹中的readme.md  2.2 GUI界面:  使用了Python的tkinte模块实现GUI 界面，大致界面如下图：  其中高级菜单如下图：    所有按钮功能如下：  附上sample:  S->S+T  T->a  T->Tb  T->T  E->F   1. 打开文件：   打开一个含有相关文法的文件，如文件夹中的samples.txt   1. 消除无用及有害规则   即消除sample中的T->T及E->F的情况  实际消除后效果如图：    （3）消除左递归：  即消除文法中的直接和间接左递归，如sample中的T->Tb , S->S+T的情况  实际效果如图：    （4）消除左公因子：  即消除文法中的左公因子，例子见测试样例  （5）生成first集：  即生成处理过的文法的first集合，使用龙书上的算法实现  实际效果如图：    （6）生成follow集：  即生成处理过的文法的follow集合，使用龙书上的算法实现  实际效果如图：    （7）生成LL(1)分析表：  即生成文法的LL（1）分析表，用于下一步的预测分析  实际效果如图：    （8）生成最左推导判断栈：  即在输入相关串后给出现有文法的最左推导过程，以龙书的分析栈算法实现  实际效果如图：     1. 实验测试：   3.1 测试样例：  选取readme.md中的部分测试样例进行测试  样例1：  S->Qc|c  Q->Rb|b  R->Sa|a  样例2：  S->S+S|S-S|S\*S|S/S| (S)  3.2 测试结果：  3.2.1 样例1：          3.2.2 样例2：          注：主程序采用pyinstaller 打包完整exe，无需额外准备电脑环境 |
| **四、实验总结**  通过本次实验，更加深入的学习了LL（1）文法及其分析功能的实现。LL(1)文法是最左推导中十分实用的一种文法，对于语法分析的自动化有着不可磨灭的作用，也是我们学习编译原理过程中的不可或缺的一环。  在实现过程中，重新设计了数据结构，决定直接设计出一个类来实现相关的功能，同样使用了tkinter作为GUI界面，十分小巧简洁。  本次实验成功实现了人为复原LL（1）分析过程的功能，但是在测试中仍存在许多的bug，很多问题以现在设计的结构来说还是没办法修复的，所以算是小小的遗憾吧。 |
| 五、参考文献：  （1）CSDN blog  （2）编译原理及实践，机械工业出版社，Kenneth C.Louden著，冯博琴 冯岚等译  （3）数据结构（用C++面向对象语言描述） 清华大学第二版 |