实验二 语法分析器

1、动机/目标

本实验的主要目标是实现一个语法分析器，构造给定文法的预测分析表。该分析器旨在确定输入的符号串是否属于该文法定义的语言。

2、内容描述

本实验涉及开发一个 C++ 程序，该程序读取一组文法产生式，计算 FIRST 集和 FOLLOW 集，构建预测分析表，并分析输入符号串以检查其是否符合文法。

3、思路/方法

1. **输入读取**：程序从用户处读取文法产生式。
2. **FIRST 集计算**：计算终结符和非终结符的 FIRST 集。
3. **FOLLOW 集计算**：计算非终结符的 FOLLOW 集。
4. **预测分析表**：基于 FIRST 和 FOLLOW 集构建预测分析表。
5. **符号串分析**：使用预测分析表分析输入符号串，以确定其有效性。

4、假设

- 提供的文法是 LL(1) 文法，且无左递归和公共前缀。
- 特殊符号：@ 表示空串，# 表示输入结束，H 用于替代 E'，J 用于替代 T'。

5、相关有限自动机描述

该 LL(1) 语法分析器的有限自动机涉及表示不同文法规则处理的状态，基于输入符号的转换，以及对应于文法符号的压栈和出栈操作。

6、重要数据结构描述

1. **向量 proce**：存储文法产生式。
2. **向量 first 和 follow**：分别存储 FIRST 集和 FOLLOW 集。
3. **映射 getnum**：将文法符号映射到唯一整数。
4. **数组 table**：表示预测分析表。
5. **栈 sta**：用于符号串分析。

7、核心算法描述

1. **FIRST 集计算**：通过检查产生式，递归计算每个非终结符的 FIRST 集。
2. **FOLLOW 集计算**：通过考虑非终结符在产生式中的位置及其与其他符号的关系，计算 FOLLOW 集。
3. **预测分析表构建**：根据 FIRST 和 FOLLOW 集填充预测分析表。
4. **符号串分析**：使用基于栈的方法模拟解析过程，验证输入符号串的有效性。

8、运行案例

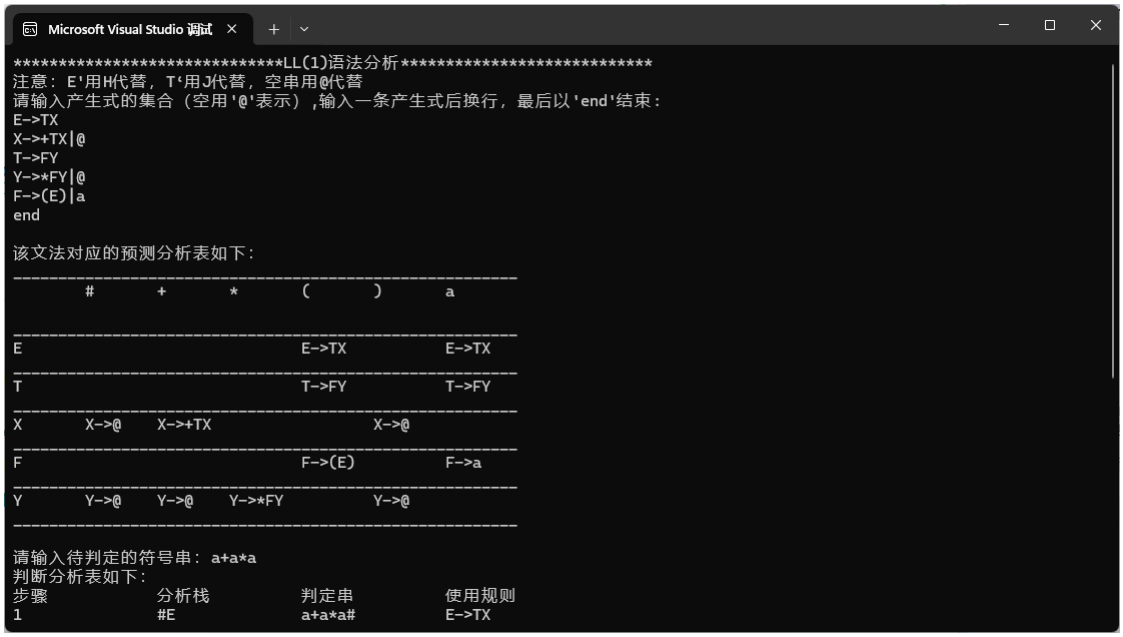
示例文法输入：

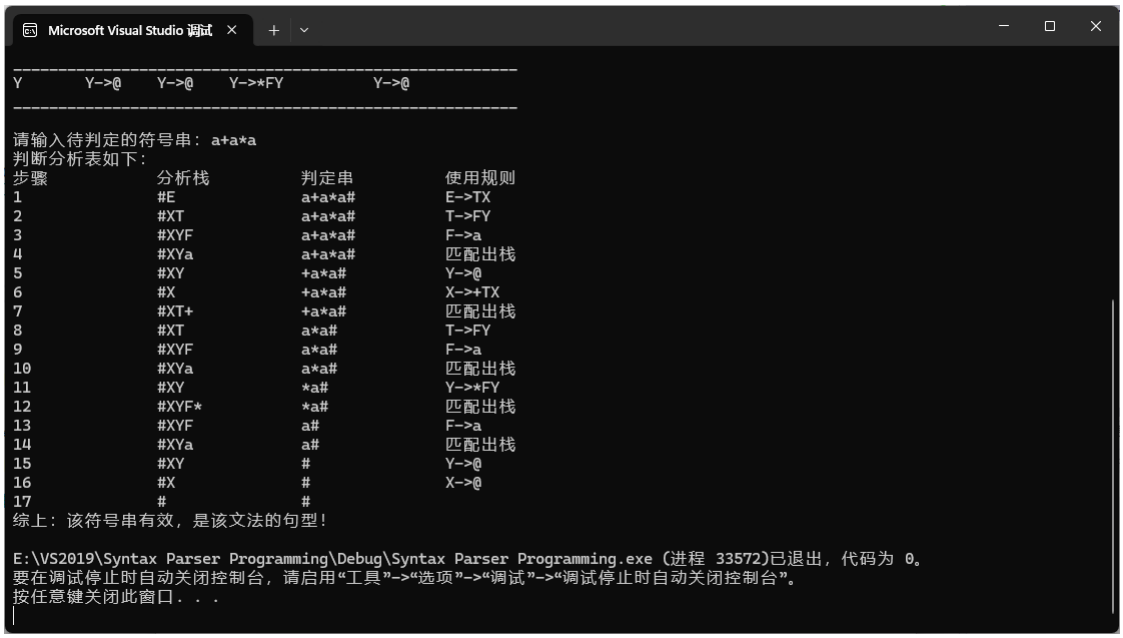
E->TX
X->+TX|@
T->FY
Y->*FY|@
F->(E)|a
end

示例输入符号串：

a+a*a

预期输出：





9、出现的问题及相关解决方案

1. **弃用警告**：freopen 函数被标记为不安全。解决方案：使用 freopen_s 或添加 _CRT_SECURE_NO_WARNINGS。
2. **文件锁定问题**：在之前运行后重新打开可执行文件时出错。解决方案：确保程序关闭后再重新编译。
3. **解析表错误**：由于 FOLLOW 集计算错误，解析表中的条目不正确。解决方案：调试并修正 FOLLOW 集逻辑。

10、个人感想与评论

本次实验提供了关于 LL(1) 解析器实现及构造预测分析表的宝贵见解。该过程加强了我对上下文无关文法和解析技术的理解。在解决弃用警告和解析表错误等问题时，提升了我在编译器设计背景下的问题解决能力。