**深 圳 大 学 实 验 报 告**

**课程名称：­ 编译原理**

**实验项目名称： 高级语言及其文法**

**学院： 计算机与软件学院**

**专业： 计算机科学与技术**

**指导教师： 张昊迪**

**报告人： 郑雨婷 学号： 2021150122 班级： 高性能**

**实验时间： 2024年 3月 15 日 -- 2024年 3 月 29 日**

**实验报告提交时间： 2024年 3月 29 日**

**教务处制**

1. 实验内容

文法（Grammar）是描述高级语言语法结构的重要工具。定义任意的文法 G， 需要完成对其四元组（V,T,P,S）的定义（课本 P33）。在该实验中，请制定文法 文件的具体组织形式、编程完成对文法文件的读取、并完成对文法的分类。该实 验具体包含以下两个任务：

* **任务一：文法的定义及读取**

现规定文法由 Grammar.txt 文件保存，请制定文法文件的具体存储格式。如 文法 G={{S,A,B,C}, {a,b,c}, {S->ABC,A->a,B->b,C->c}, S}在 Grammar.txt 文件 中可由以下方式描述并存储：

--------------------------------------------------

S,A,B,C

a,b,c

S->ABC,A->a,B->b,C->c

S

--------------------------------------------------

文法的文本形式可根据自己需要自由定义，在此基础上，编程实现对任意文 法文件的读取。

* **任务二：文法的分类**

根据 Chomsky 的文法体系分类（课本 P40），文法分为四大种类。请在任务 一的基础上，编程实现对 Grammar.txt 中存储的文法进行分类，自动判断其所属 类别。例如任务一中所给出的文法 G 应被判定为 2 型文法，即上下文无关文法。 请设计分类方法，并设计四类不同的测试文法测试分类结果的正确性。

二、实验要求

* 使用 C、C++、或 Java 等语言完成任务一、二的程序编写；
* 使用实验所提供的模板撰写实验报告，要求内容详实，有具体的设计描述、 关键的代码片段、及实验结果屏幕截图；
* 在截止日期前将 ppt、代码、实验报告、测试文件等所有实验相关文件压缩 到一个压缩包姓名\_学号\_实验一.zip 上传至 Blackboard。

三、实验步骤

1. 文法的定义及读取

我选择使用Java语言来完成本次实验。首先定义文法的文本形式如下：

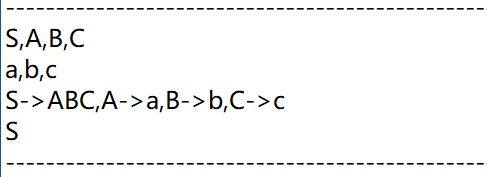


图1 定义文法的文本形式

接下来创建Grammar类，根据文法的定义，可以定义V、T、P、S四个静态成员变量，其中V是非终结符集、T是终结符集、P是产生式集、S是开始符号。

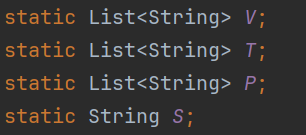


图2 声明V、T、P、S四个成员变量

分析文法的文本形式，第一行是分割线无意义，第二行是非终结符集V、第三行是终结符集T、第四行是产生式集P、第五行是开始符号S、最后一行是分割线无意义。可以发现每一行是什么都已经固定好，因此我们按行读取之后再进行处理。

Java语言提供了对文件内容逐行读取的方式，使用BufferedReader的line()方法，即可读取一行。BufferedReader构造函数接受一个Reader实例（如FileReader）作为字符输入流源，因此先构造一个FileReader对象，其中的pathname是文法文件的绝对地址。

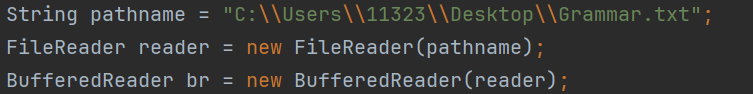


图3 获取输入流

对于前后两行的分割线，可以接收了但不做任何处理。

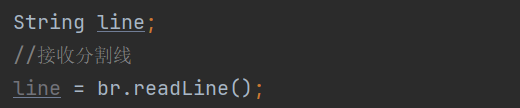


图4 按行接收

中间四行的需要进行处理。首先读取到非终结符V所在的这一行后，可以看到每两个变量之间是用逗号分隔开的，Java的String类型提供了split方法，可以将一个字符串按照给定的字符串分割，因此直接line.split(“,”)就可以返回一个String数组，将其转变成List后存入成员变量V中。下一行是终结符集T所在的行，与V相同，按照逗号分割后存入成员变量T。下一行是P，也是相同的操作，但是值得注意的是，每一个产生式可能是多个左部相同的产生式的合体，虽按照形式存入了List集合，但是在后续处理时应该注意。最后的开始符合是单一的字符，无需分割，直接存入成员变量S。

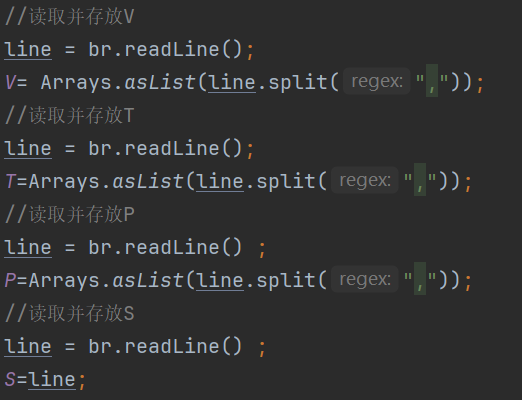


图6 文件中的四行分别对应V、T、P、S

至此，文法文件的读取已经结束，编写代码来验证正确性。将V、T、P、S输出到控制台，将V、T、P循环输出，每个元素之间用空格分开，S是一个开始符号，只有一个，直接输出。

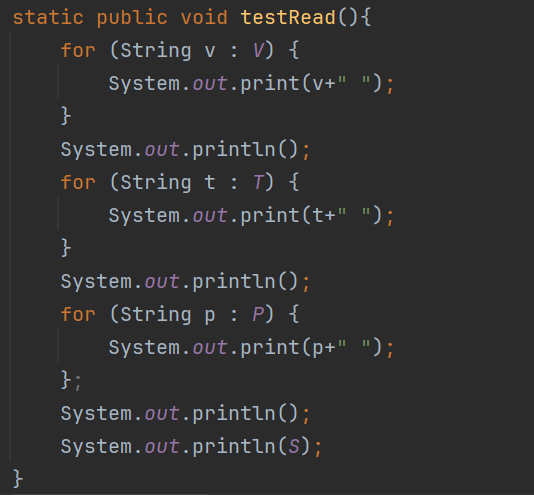


图7 设计测试读取代码

2.文法的分类

（1）判断是否为0型文法

首先需要判断文件中的文法是否满足文法的定义，所以我设置文法的类型type初始为-1。第一，文法是开始符号必须是非终结符V的元素，否则它不构成文法，返回-1；第二，文法的开始符号只能有一个，但我是默认开始符号的那一行只有一个元素，当作一个字符串读取进来的，所以判断有无逗号就可以。第三，终极符集和非终结符集的交集必须为空，若不为空，不构成文法。

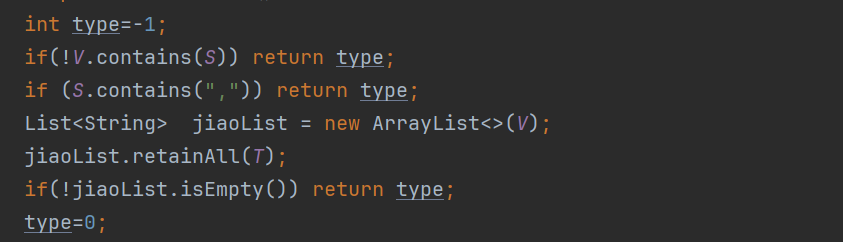


图8 判断是否构成文法

（2）判断是否为1型文法

在是0型文法的基础上，要成为1型文法，需要每一条产生式的右部的长度大于等于左部的长度。因此需要遍历产生式集P的每一条产生式，判断是否符合条件，一旦出现左部长度大于右部的，就不符合1型文法的条件，返回当前type（值为0）。但还有一点需要注意，产生式的右部可能是多个左部相同的产生式的合体，所以需要分情况讨论，当右部不包含分割线 | 时，直接比较左部与右部的长度；当右部包含 | 时，需要用spilt将右部分开再分别比较。若遍历完成，没有不符合条件的设type=1。

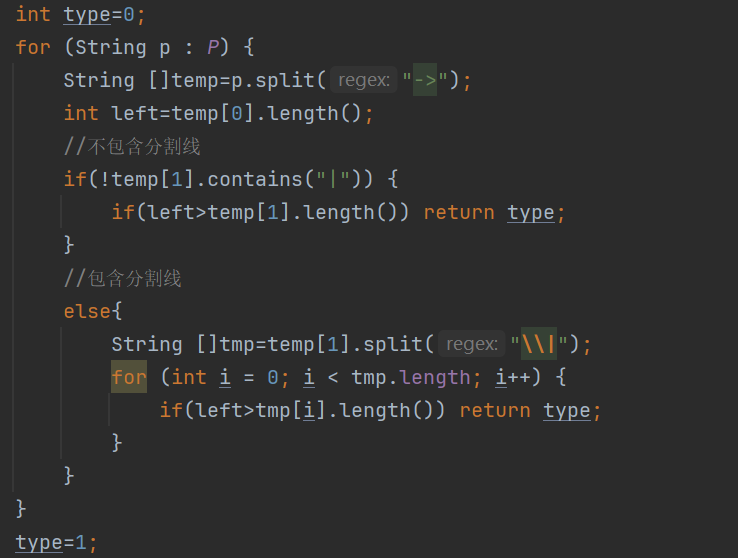


图9 判断是否满足1型文法

（3）判断是否为2型文法

在是1型文法的基础上，要成为2型文法，需要每一条产生式的左部都属于非终结符集V。因此需要遍历产生式集P的每一条产生式，判断是否符合条件，一旦出现左部不属于非终结符集的，就不符合2型文法的条件，返回当前type（值为1）。若遍历完成，没有不符合条件的设type=2。

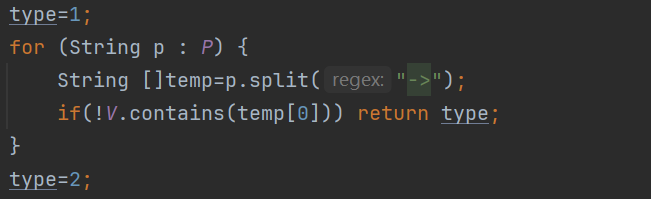


图10 判断是否满足2型文法

（4）判断是否为3型文法

在是2型文法的基础上，要成为3型文法，需要是右线性文法或者左线性文法，但不能左、右线性混用，所以，类中再增加两个静态成员变量Left和Right表示是左还是右线性。遍历产生式集P的每一条产生式，判断是否符合条件，一旦出现既不是左线性文法又不是右线性文法的，就不符合3型文法的条件，返回当前type（值为2）。具体的，如果一个产生式右部既不是单一的终结符、又不符合wA的形式、也不符合Aw的形式，就返回type。

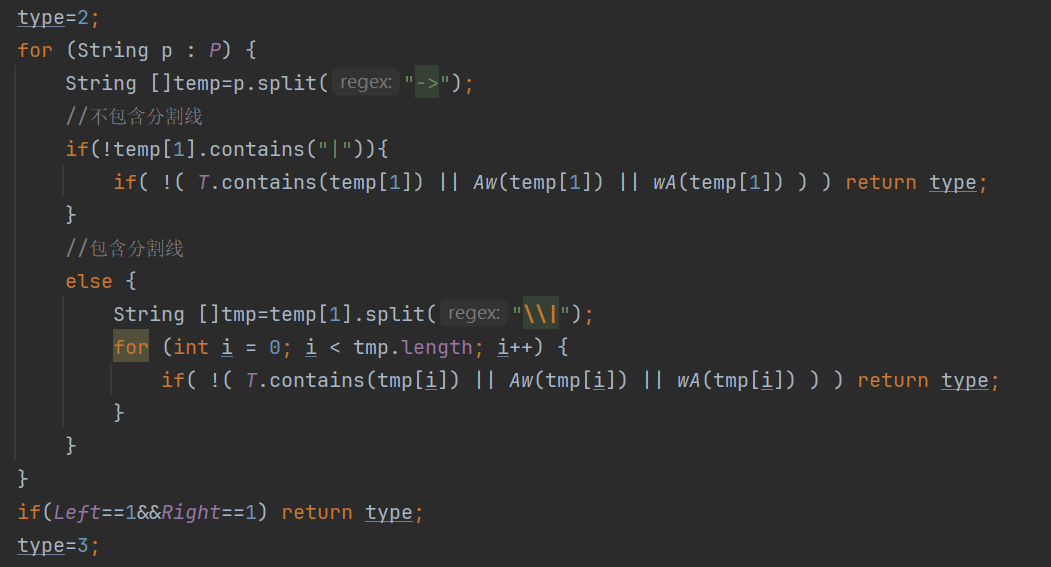
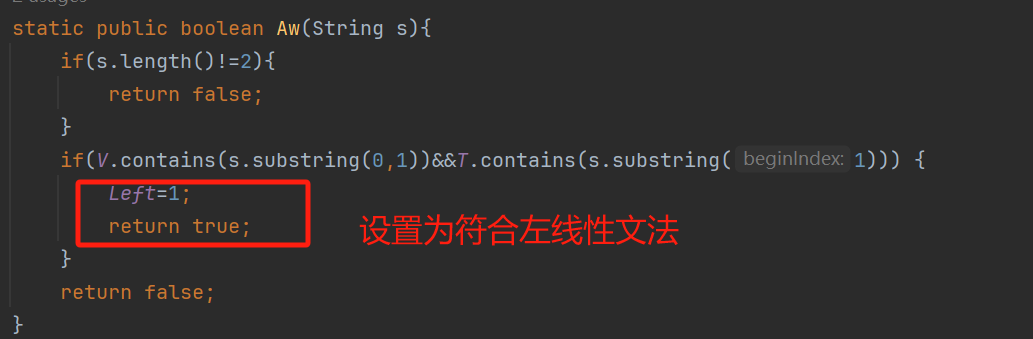


图11 判断是否满足3型文法

其中，判断是否符合Aw的形式的代码如下,首先长度一定为2，然后截取第一个字符必须是属于非终结符的并且第二个字符必须是属于终结符的 。相同的，判断是否符合wA的形式的代码如下,首先长度一定为2，然后截取第一个字符必须是属于终结符的并且第二个字符必须是属于非终结符的 。

如果遍历完所有产生式了，不能直接认定就是3型文法了，还需判断有无左右线性文法混用。在判断出某个产生式符合A->wB的形式，就将Right置为1；相同的，在判断出某个产生式符合A->Bw的形式，就将Left置为1。这样，在遍历所有产生式结束后，判断Left和Right不同为1才能将type设为3。



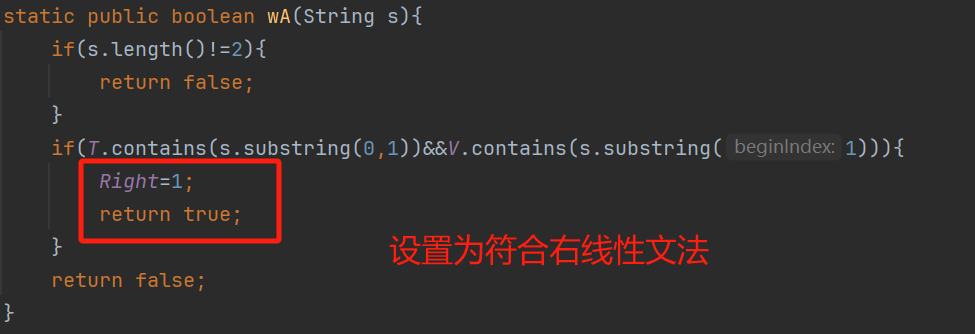


图12 左、右线性文法不能混用

四、实验结果

1. 文法的定义及读取

将输出结果与原本的文法文件进行对比，结果一致，读取文件正确。

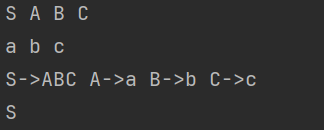
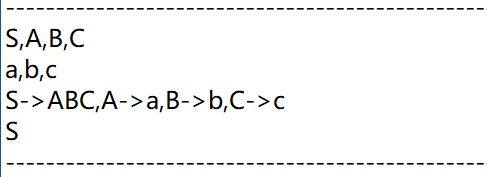
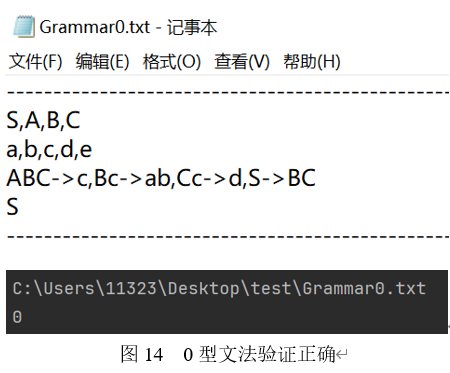
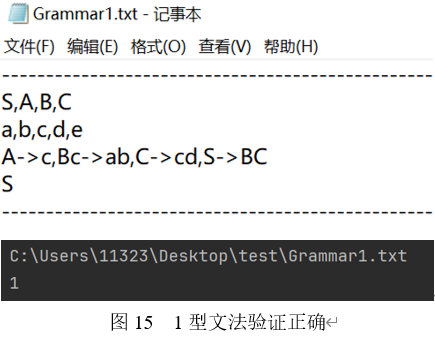
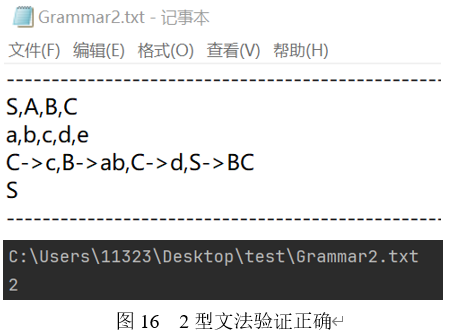
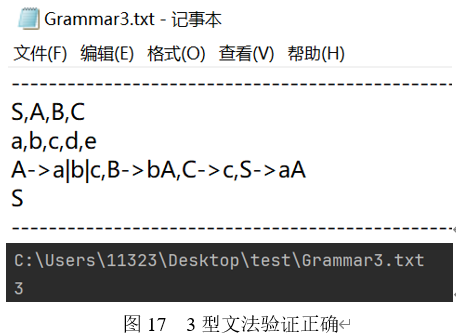
 

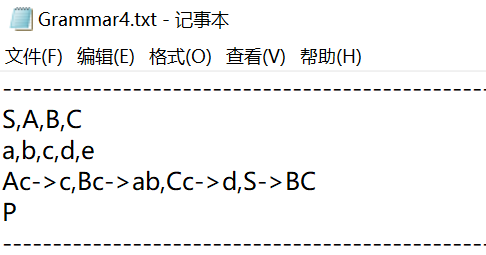
图13 文法读取验证正确

2. 文法的分类

设计五个不同的测试文件grammar0.txt、grammar1.txt、grammar2.txt、grammar3.txt、grammar4.txt分别对应0型文法、1型文法、2型文法、3型文法、不是文法。判断结果如下：



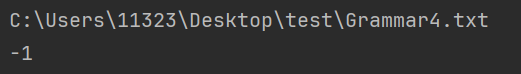


图18 非文法验证正确

五、实验总结

本次实验目的主要是加深对文法概念的理解。首先，我定义了文法的存储格式，并编写了程序实现了对文法文件的读取。文法文件的格式按照给定的要求进行定义，包括非终结符集合、终结符集合、产生式规则以及起始符号。通过程序，可以轻松地读取文法文件中的信息，并将其存储在程序中。

之后，我实现了文法的分类功能。通过检查文法产生式规则的形式和结构，我将文法准确地分类到相应的类型中。并且我设计了多组测试文法，包括各种类型的文法以及一些边界情况，以验证程序对文法的分类是否准确。通过运行测试文法，我们确认了程序能够正确地对文法进行分类，并得出了符合预期的结果。

总之，通过本次实验，我对文法的概念以有了更深的理解。

|  |
| --- |
| **指导教师批阅意见：**  **成绩评定：**  **指导教师签字：**  年 月 日 |
| 备注： |