nodejs语法分析器实验报告

**{ 班级 : 软开一班**

**成员一 : 葛高强**

**学号 : 180310325**

**成员二 : 吴巧巧**

**学号 : 180310314**

**}**

**一．实验目的**

**我们使用nodejs作为所使用的高级语言完成对于一个给定的LL(1)文法，实现自顶向下的预测分析方法，逐步构造其预测分析表，并对于指定的输入串进行分析判别，输出详细的分析过程。**

**二．实验要求**

**使用一门高级语言来对于给定文法进行分析，更加了解语法分析的过程，使用高级语言实现自动化分析流程加深了我们对于语法分析的印象，并且更加熟悉运用自己掌握的高级语言。**

**三．实验流程**

**// js语法分析器模块**

**// === 我的理解过程 ===**

**/\*首先实现集合FIRST(X)构造算法和集合FOLLOW(A)构造算法，再根据FIRST和FOLLOW集合构造出预测分析表，并对指定的句子打印出分析栈的分析过程，判断是否为该文法的句子。但是现在只是实现了构造预测分析表，并进行分析。\*/**

**// 语法分析器对js语言进行最初步的解析操作**

**/\*（1）求FIRST集的算法思想**

如果产生式右部第一个字符为终结符，则将其计入左部first集

 如果产生式右部第一个字符为非终结符

 ->求该非终结符的first集

 ->将该非终结符的非$first集计入左部的first集

->若存在$，则将指向产生式的指针右移

->若不存在$，则停止遍历该产生式，进入下一个产生式

->若已经到达产生式的最右部的非终结符，则将$加入左部的first集

 处理数组中重复的first集中的终结符

**（2）求FOLLOW集的算法思想**

 对于文法G中每个非终结符A构造FOLLOW(A)的办法是,连续使用下面的规则,直到每个FOLLOW不在增大为止.

(1) 对于文法的开始符号S,置#于FOLLOW(S)中;

(2) 若A->aBb是一个产生式,则把FIRST(b)\{ε}加至FOLLOW(B)中;

(3) 若A->aB是一个产生式,或A->aBb是一个产生式而b=>ε(即ε∈FIRST(b))则把FOLLOW(A)加至FOLLOW(B)中

**（3）生成预测分析表的算法思想**

构造分析表M的算法是：

 (1) 对文法G的每个产生式A->a执行第二步和第三步;

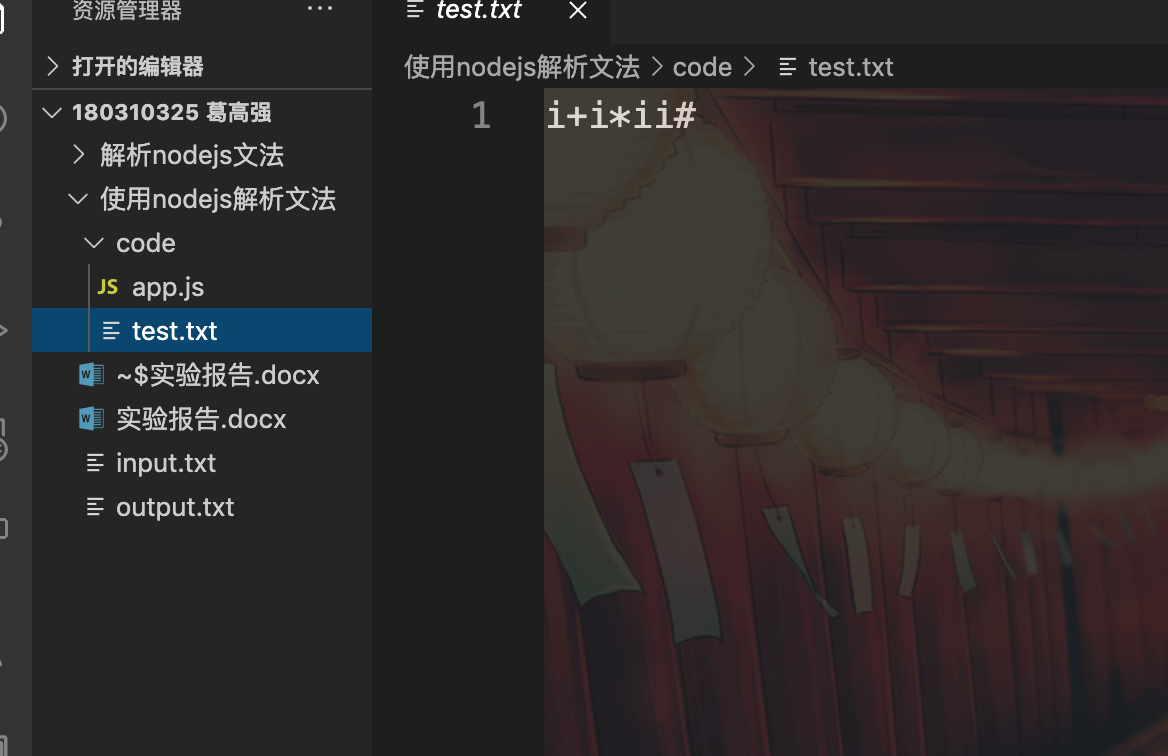
 (2) 对每个终结符a∈FIRST(a),把A->a加至M[A,a]中;

 (3) 若ε∈FIRST(a),则把任何b∈FOLLOW(A)把A->a加至M[A,b]中;

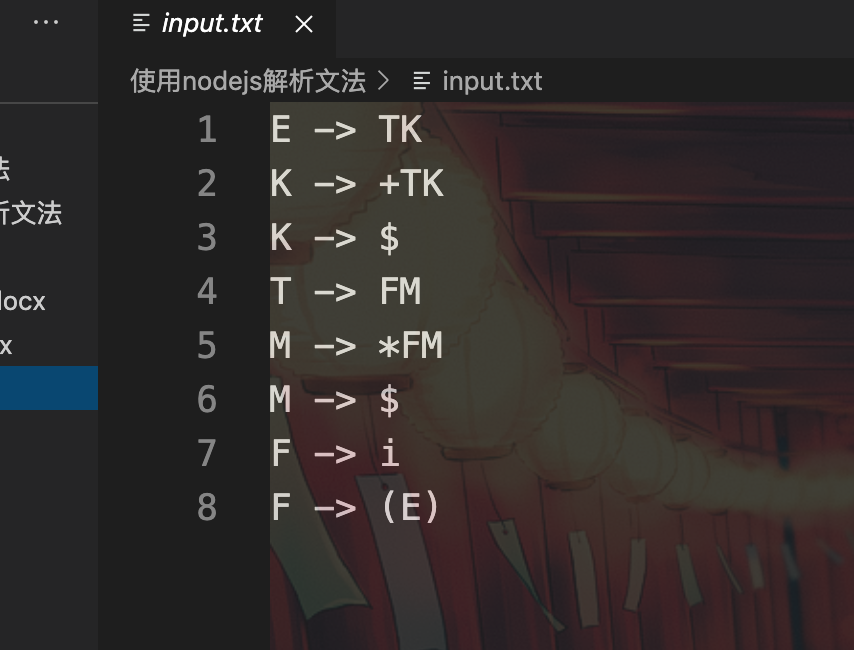
 (4) 把所有无定义的M[A,a]标上出错标志.\*/

**四．实验结果图**

**1). 构造使用的匹配的语句，在test.txt文件之中**

****

**2）. 构造输入的文件**

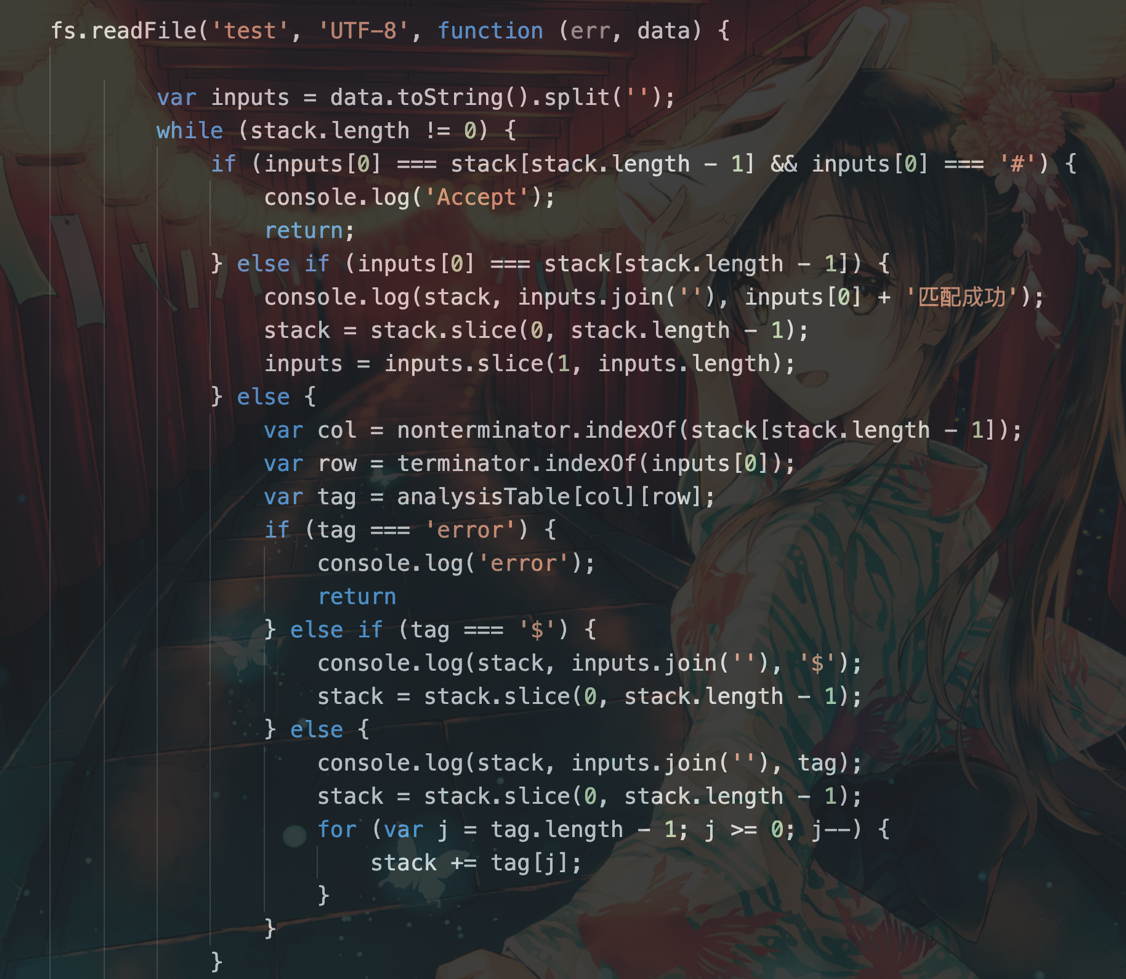
****

**3）. 分析结果(输出到了output文件里面)**

****

**五．关键代码分析**

**1）. 使用栈结构对于分析过程更加友好，给予用户错误的代码提示。**

****

**2）. 构造特定的语法分析表**

****

**六．心得体会**

**认识到自己对知识的理解的不充分，想了很久没有想到如何使用自己的语言构造语法分析器，认识到自己的代码能力有待提示，有参考相关网上的解析，学习到了很多的关于语法分析器的知识，收货颇多。**