第三次实验

实验目的

开发一个完整的PL/0编译器

1. 文件输入：符合PL/0文法的源程序（自己要有5个测试用例，包含出错的情况，还要用老师提供的测试用例进行测试）
2. 输出：P-Code
3. 错误信息：参见教材第316页表14.4（新书第411页表17.4）。
4. 错误信息尽量详细（行号，错误类型）
5. P-Code指令集：参见教材第351页表15.14。
6. 语法分析部分要求统一使用递归下降子程序法实现。

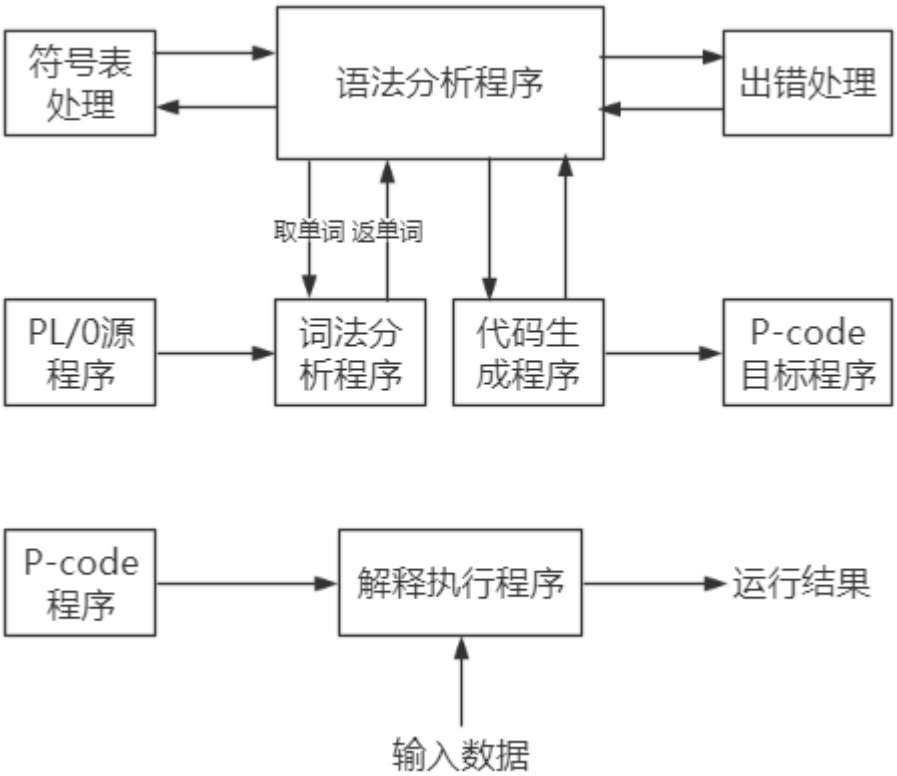
实验步骤

文法

```
<程序> ::= <分程序>.
<分程序> ::= [<常量说明部分>][变量说明部分>][<过程说明部分>]<语句>
<常量说明部分> ::= const<常量定义>{,<常量定义>};
<常量定义> ::= <标识符>=<无符号整数>
<无符号整数> ::= <数字>{<数字>}
<标识符> ::= <字母>{<字母>|<数字>}
<变量说明部分>::= var<标识符>{,<标识符>};
<过程说明部分> ::= <过程首部><分程序>;{<过程说明部分>}
<过程首部> ::= procedure<标识符>;
<语句> ::= <赋值语句>|<条件语句>|<当型循环语句>|<过程调用语句>|<读语句>|<写语句>|<复合语句>|<重复语句
>|<空>
<赋值语句> ::= <标识符>:=<表达式>
<表达式> ::= [+|-]<项>{<加法运算符><项>}
<项> ::= <因子>{<乘法运算符><因子>}
<因子> ::= <标识符>|<无符号整数>|'('<表达式>')'
<加法运算符> ::= +|-
<乘法运算符> ::= *|/
<条件> ::= <表达式><关系运算符><表达式>|odd<表达式>
<关系运算符> ::= =|<>|<|<=|>|>=
<条件语句> ::= if<条件>then<语句>[else<语句>]
<当型循环语句> ::= while<条件>do<语句>
<过程调用语句> ::= call<标识符>
<复合语句> ::= begin<语句>{;<语句>}end
<重复语句> ::= repeat<语句>{;<语句>}until<条件>
<读语句> ::= read'('<标识符>{,<标识符>}')'
<写语句> ::= write'('<标识符>{,<标识符>}')'
<字母> ::= a|b|...|X|Y|Z
<数字> ::= 0|1|2|...|8|9
```

PL/0系统结构

PL/0编译系统是一个编译-解释执行程序，整个编译过程分两个阶段进行。第一阶段先把PL/0源程序编译成假想计算机的目标程序（P-code指令），第二阶段再对该目标程序进行解释执行，得到运行结果。PL/0编译程序采用一遍扫描，即以语法分析为核心，由它调用词法分析程序取单词，在语法分析过程中同时进行语义分析处理，并生成目标指令。如遇到语法、语义错误，则随时挑用出错处理程序，打印出错信息。在编译过程中要利用符号表的登录和查找来进行信息之间的联系。一遍扫描的PL/0编译和P-code解释执行框图如下图所示。



词法分析

- KEYWORD 关键词
- INDENTIFIER 标识符
- DELIMITER 分界符
- SINGLE_OPERATOR 单字符运算符
- DOUBLE_OPERATOR 双字符运算符
- NUMBER 数字

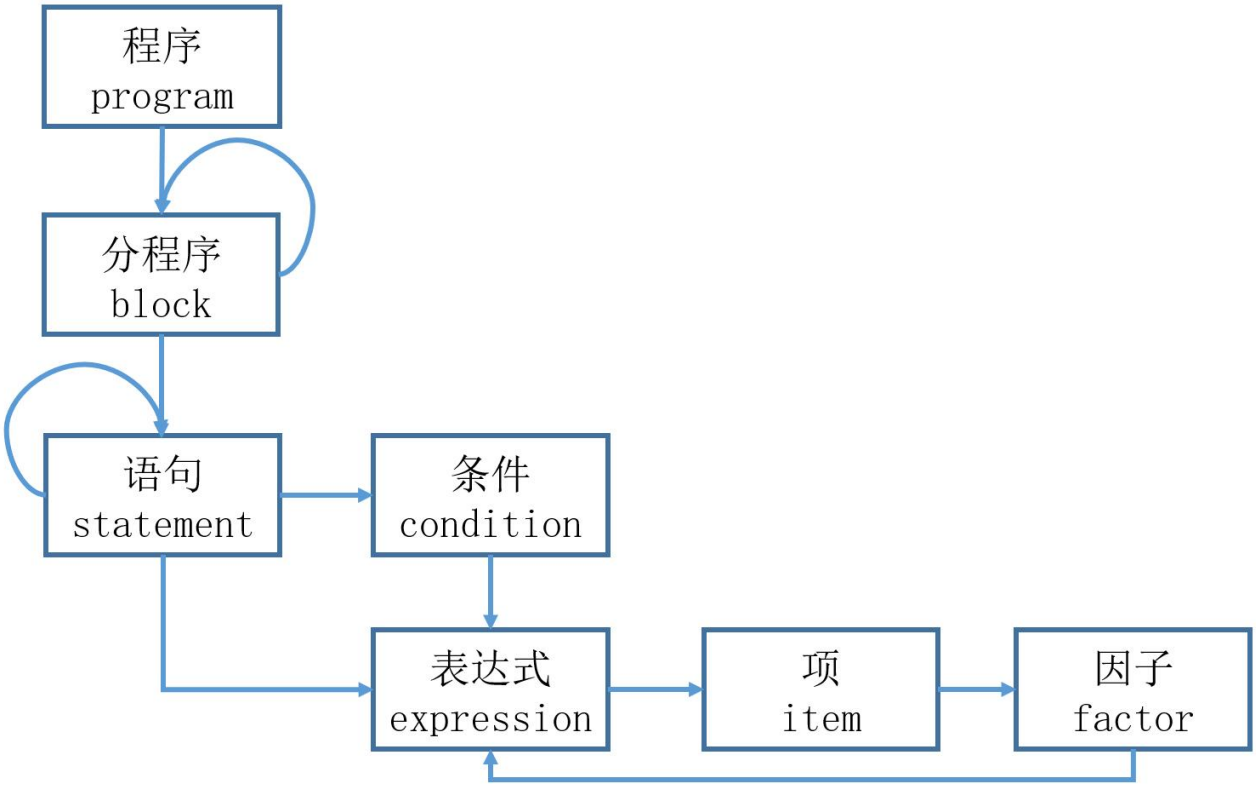
符号表管理

- name 名字
- kind 类型 **const/var/proc**
- val 值 **cont专用**
- level 层次 **var/proc专用**
- adr 地址 **var/proc专用**

在常量、变量、过程声明时进行**登记**，使用时从符号表末尾往前**查询**

语法分析

PL/0采用了**递归子程序法**进行语法分析，即为每一个语法成分编写一个分析子程序，根据当前读取的符号，可以选择相应的子程序进行语法分析。



生成P-Code

Pcode语言：一种栈式机的语言。此类栈式机没有累加器和通用寄存器，有一个栈式存储器，有四个控制寄存器（指令寄存器 I，指令地址寄存器 P，栈顶寄存器 T和基址寄存器 B），算术逻辑运算都在栈顶进行。

指令格式形如**F L A**，其中**F**表示操作码，可用枚举变量表示；**L**表示层次差 （标识符引用层减去定义层）；**A**不同的指令含义不同。

在**语法分析**的过程中，同时也在不断地生成Pcode，在语法分析函数中调用函数将指令写入code指令数组中。

对于if then [else]，while do和repeat until语句，要生成跳转指令，采用**地址返填**技术。目标代码生成模式如下：

```
 (1)  if <condition> then <statement> [else]
<condition>
JPC addr1
<statement>
addr1:
[JMP addr2
<statement>
addr2:]
 (2)  while <condition> do <statement>
addr1:<condition>
JPC addr2
<statement>
JMP addr1
addr2:
 (3)  repeat <statement> until <condition>
addr1:<statement>

<condition>
```

```
JPC addr1
```

错误处理

在语法分析的时候进行错误处理

出错编号	出错原因
1	常数说明中应是'='而不是':='
2	常数说明中'='后应为数字
3	常数说明中标识符后应为'='
4	const,var,procedure后应为标识符
5	漏掉逗号或分号
6	过程说明后的符号不正确
7	应为语句开始符号
8	程序体内语句部分后的符号不正确
9	程序结尾应为句号
10	语句之间漏了分号
11	标识符未说明
12	不可向常量或过程赋值
13	赋值语句中应为赋值运算符':='
14	call后应为标识符
15	call后标识符属性应为过程,不可调用常量或变量
16	条件语句中缺失then
17	应为分号或end
18	当到型循环语句中缺失do
19	语句后的符号不正确
20	应为关系运算符
21	表达式内不可有过程标识符
22	缺失右括号
23	因子后不可为此符号
24	表达式不能以此符号开始
25	循环语句中缺失until
26	读语句括号内不是标识符
27	写语句括号内不是表达式

出错编号	出错原因
28	读语句括号内标识符属性应该为变量
29	标识符类型不正确
30	这个数太大，超过INT32_MAX
31	标识符重名
32	嵌套层数过多
33	句号后程序未结束
34	应为整数而不是浮点数
40	应为左括号
50	无法识别的符号

程序句号后面还有跳过并声明错误

程序设计

1. 符号表
    - 结构
    - 登记
    - 查询
2. 指令记录表
    - 结构
    - 登记
    - 打印
3. P-Code生成器——语法分析中包含多个递归子程序，取词法分析器的单次进行语法分析，同时生成指令，并对错误进行处理
    - block

      处理常量声明、变量声明和过程声明，调用statement子程序
    - statement

      处理赋值、条件、当循环、过程调用、复合、重复、读和写语句，赋值语句会调用expression子程序，条件、当循环和重复语句会调用condition和expression子程序，重复语句调用statement子程序（自身），写语句调用expressiong子程序
    - condition

      调用expression子程序
    - expression

      调用item子程序
    - item

      调用factoe子程序

- factor

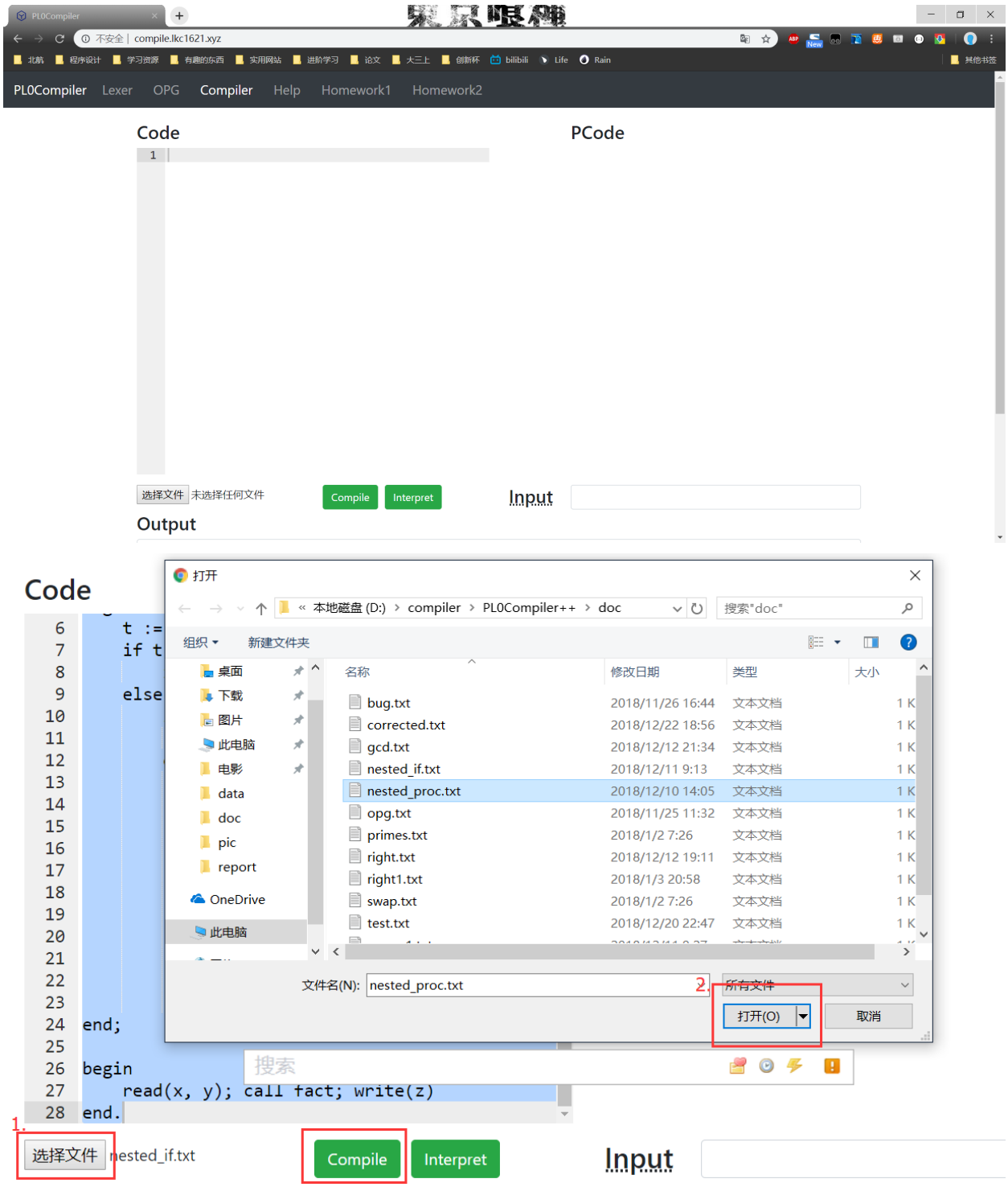
  调用expression子程序

操作

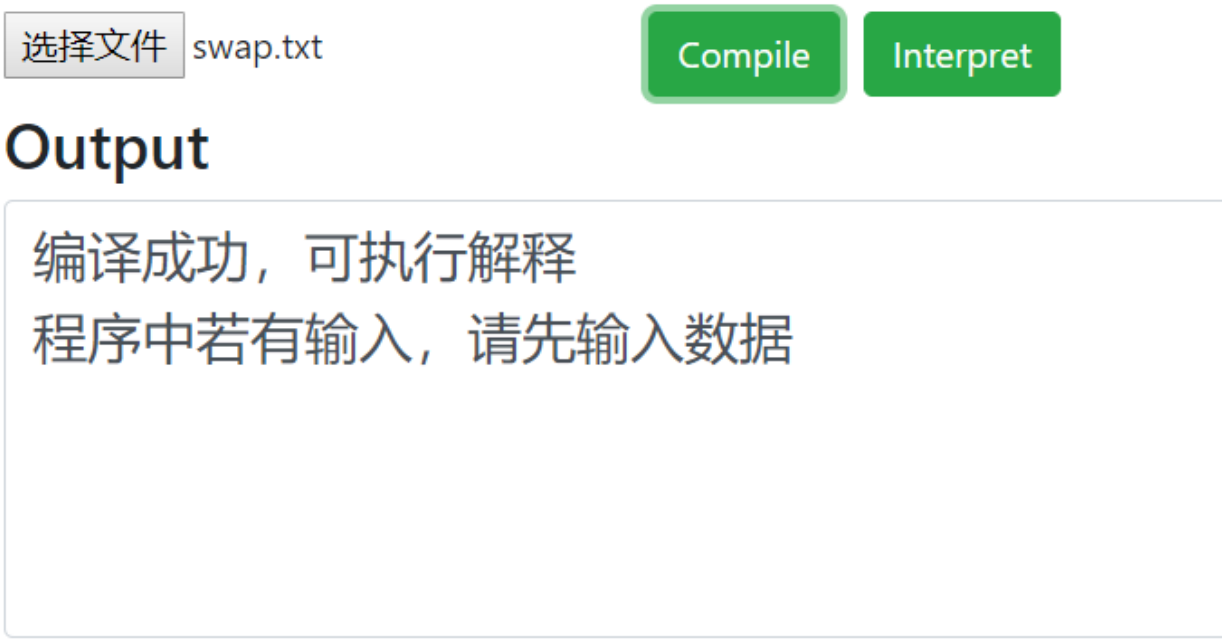
程序可在网页进行访问http://compile.lkc1621.xyz/

导航栏上有三个功能栏，Lexer是词法分析器，OPG是算符优先程序，Compiler是编译器，而Help为一些帮助信息，Homework1和Homework2是前两次的作业，默认进入的是编译器界面。

可直接在Code中输入源代码，也可选择文件输入。

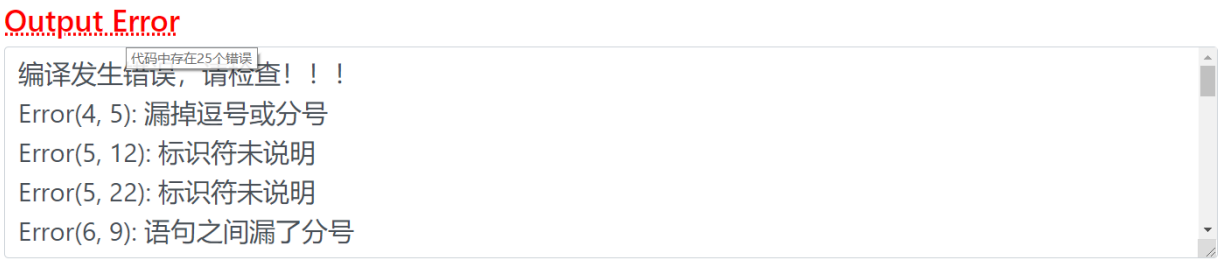


点击Compile进行程序编译，若提示编译成功，可进行代码解释。



点击Interpret进行解释，若未输入网页会弹出对话框提示输入，输入完成后重新点金Interpret进行解释。

若提示编译失败，需要根据错误信息进行代码修改，之后重新编译。



实验感想

整个编译器，前前后后花了大概两周的时间，一开始毫无思路，认认真真地看了书上第十四章的内容大概明白了整个程序的过程，再细心体会书上附录的源代码，才开始着手写程序。

由于之前对PL/0语法进行过词法分析，在这一部分自己倒不遇到很大的问题，但有所改变的是这一次并不是整个代码一次全部分析完，而是通过语法分析程序一个一个取得单词，作为语法分析的资源池，因而首先需要修改词法分析程序，完善单词的类型和输出。

之后最难的一块莫属于语法分析模块，语法分析由递归子程序构成，还连接着P-code生成和错误处理，而错误处理中最难的莫过于错误跳读。自己在完成语法分析时，首先未考虑到错误跳读，直接书写递归子程序生成P-code，在错误发生时直接进行raise error处理，花了两天的时间完成了基本的编译程序。但由于编译在发生一个错误时便会停止，之后便把错误信息记录在表中。之后是进行跳读处理，这里要充分考虑每个子程序的开始符号集合和合法的

停止符号集合，在程序调用前进行判断，若出现错误则寻找下一个合法的符号并进行错误提示。这里道理简单，但涉及程序时需要小心谨慎地考虑每一种情况，这里花费了自己好多天的时间，不断地测试修改bug。至于P-code的解释程序，个人觉得比较简单，每个指令都有严格的意义，理解原理便可快速完成。

另外一个花时间的就是GUI界面的设计了，自己采用网页实现，尽可能地选择了舒适的排版，每个按钮与表格的布局都尝试了很多遍，强迫症的审美。

通过本次的时间，最大的提升就是对编译器的理解，明白了词法分析、语法分析、出错处理、符号表之间的关系，再者也提高了自己对python的编程能力，增强了类封装的意识，也从零开始学会了用flask搭建网站。最后，感谢邵兵老师一个学期一来的悉心指导，也感谢助教的解答，望身体健康，天天开心~~