PL0文法编译器文档

**概述：**

本程序是使用C#编写的PL/0语言编译器。可以对PL0代码进行编译、生成P-Code、运行。用户可以新建文件编写PL0代码，或打开已有的PL0代码进行编辑并保存。

UI界面使用WinForm实现，错误处理部分参考了《编译原理及编译程序构造》第316页表14.4，P-Code指令集参考第351页表15.14，语法分析部分使用递归下降子程序法实现，符号表管理部分使用一个模拟的栈来进行管理。

**Error list：**

errors.Add(1, "常数说明中的\"=\"写成\":=\"");

errors.Add(2, "常数说明中的\"=\"后应是数字");

errors.Add(3, "常数说明中标识符后应是\"=\"");

errors.Add(4, "const, var, procedure 后应是标识符");

errors.Add(5, "漏掉了\',\' 或\';\'");

errors.Add(6, "过程说明后的符号不正确(应是语句开始符,或过程定义符");

errors.Add(7, "应是语句开始符");

errors.Add(8, "程序体内的语句部分的后跟符不正确");

errors.Add(9, "程序结尾丢了句号\'.\'");

errors.Add(10, "语句之间漏了\';\'");

errors.Add(11, "标识符未说明");

errors.Add(12, "赋值语句中, 赋值号左部标识符属性应是变量");

errors.Add(13, "赋值语句左部标识符后应是赋值号\':=\'");

errors.Add(14, "call 后应为标识符");

errors.Add(15, "call 后标识符属性应为过程");

errors.Add(16, "条件语句中丢了\'then\'");

errors.Add(17, "丢了\'end\' 或\';\'");

errors.Add(18, "while 型循环语句中丢了\'do\'");

errors.Add(19, "语句后的符号不正确");

errors.Add(20, "应为关系运算符");

errors.Add(21, "表达式内标识符属性不能是过程");

errors.Add(22, "表达式中漏掉右括号\')\'");

errors.Add(23, "因子后的非法符号");

errors.Add(24, "表达式的开始符不能是此符号");

errors.Add(25, "repeat 型循环语句中没有until");

errors.Add(26, "程序层次结构超过限制");

errors.Add(30, "数位太长");

errors.Add(31, "数越界");

errors.Add(32, "read语句括号中的标识符不是变量");

errors.Add(33, "语句漏掉左括号\'(\'");

errors.Add(34, "语句漏掉右括号\')\'");

errors.Add(35, "read语句缺乏变量");

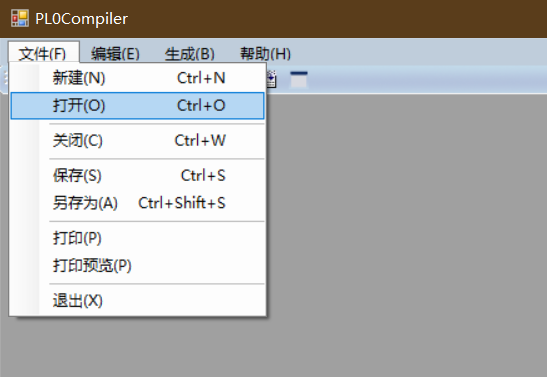
errors.Add(36, "程序体之外出现字符");

**pCpde list：**

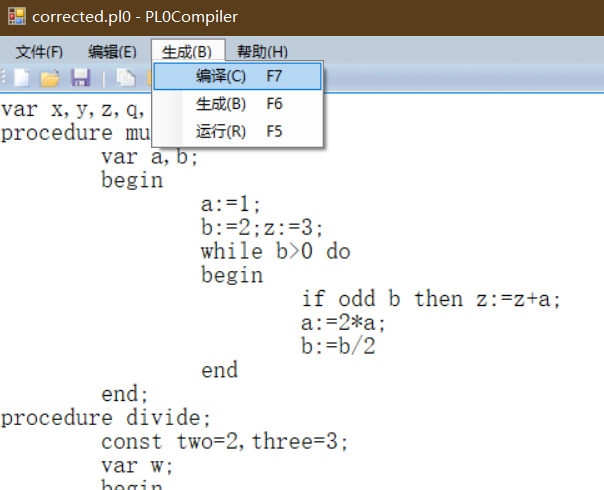
lit, lod, sto, cal, Int, jmp, jpc, opr

**测试流程（使用方法）：**

先点击菜单栏的文件按钮，选择打开一个PL/0源代码文件。



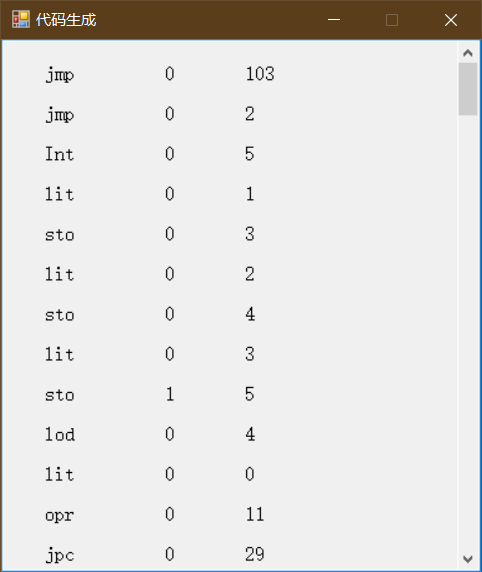
然后点击生成按钮下拉选择编译：



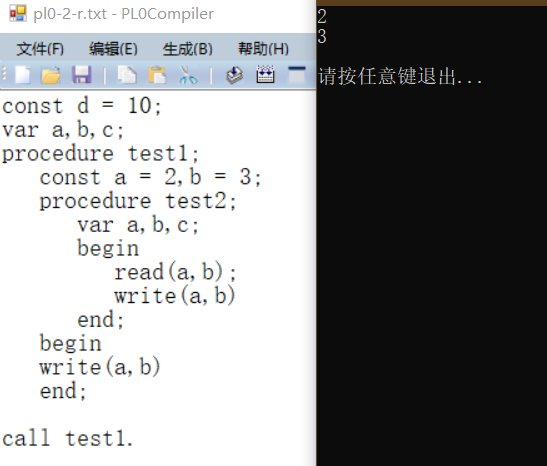
可以看到编译结果（如果有错误会提示错误类型和位置）：



之后点击生成按钮下拉选择生成，就会生成相应的P-Code：



最后点击生成按钮下拉选择运行可看到运行的结果：



**感想总结：**

本次PL/0编译器可以说是我上大学以来独立完成的程序中最复杂的一个，通过这次编写PL/0编译器，我的编码能力有了较大提升。并且通过动手实践，对编译原理这门课的学习也有了更深刻的理解。