

编译原理课程报告

162210107蔡蕾



2024-11-18

南京航空航天大学

计算机科学与技术学院

目录

[总任务 1](#_Toc180162382)

[上机练习1 1](#_Toc180162383)

[上机练习2 2](#_Toc180162384)

[系统设计 2](#_Toc180162385)

[系统结构 2](#_Toc180162386)

# 总任务

## 上机练习1

**词法分析**

**一个PASCAL语言子集（PL/0）词法分析器的设计与实现**

**PL/0语言的BNF描述（扩充的巴克斯范式表示法）**

**<prog> → program <id>；<block>**

**<block> → [<condecl>][<vardecl>][<proc>]<body>**

**<condecl> → const <const>{,<const>};**

**<const> → <id>:=<integer>**

**<vardecl> → var <id>{,<id>};**

**<proc> → procedure <id>（[<id>{,<id>}]）;<block>{;<proc>}**

**<body> → begin <statement>{;<statement>}end**

**<statement> → <id> := <exp>**

**|if <lexp> then <statement>[else <statement>]**

**|while <lexp> do <statement>**

**|call <id>（[<exp>{,<exp>}]）**

**|<body>**

**|read (<id>{，<id>})**

**|write (<exp>{,<exp>})**

**<lexp> → <exp> <lop> <exp>|odd <exp>**

**<exp> → [+|-]<term>{<aop><term>}**

**<term> → <factor>{<mop><factor>}**

**<factor>→<id>|<integer>|(<exp>)**

**<lop> → =|<>|<|<=|>|>=**

**<aop> → +|-**

**<mop> → \*|/**

**<id> → *l*{*l*|d} （注：*l*表示字母）**

**<integer> → d{d}**

**注释：**

**<prog>：程序 ；<block>：块、程序体 ；<condecl>：常量说明 ；<const>：常量；**

**<vardecl>：变量说明 ；<proc>：分程序 ； <body>：复合语句 ；<statement>：语句；**

**<exp>：表达式 ；<lexp>：条件 ；<term>：项 ； <factor>：因子 ；<aop>：加法运算符；**

**<mop>：乘法运算符； <lop>：关系运算符**

**odd：判断表达式的奇偶性。**

**要求：**

1. **按照P45的算法思想，使用循环分支方法实现PL/0语言的词法分析器，该词法分析器能够读入使用PL/0语言书写的源程序，输出单词符号串及其属性到一中间文件中，具有一定的错误处理能力，给出词法错误提示（需要输出错误所在的行列）。**

## 上机练习2

**语法分析**

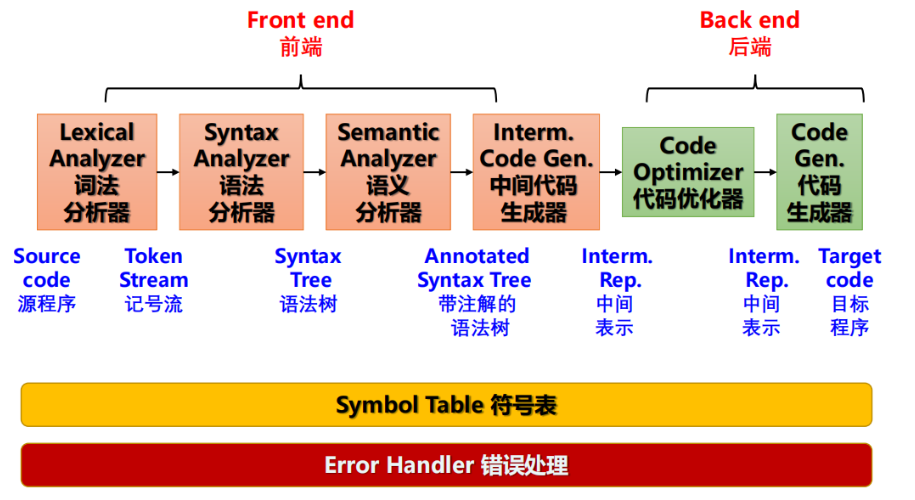
根据上机练习一给出的PL/0语言扩充的巴克斯范式语法描述，利用递归下降的语法分析方法，编写PL/0语言的语法分析程序。

要求：

1. 对给出的PL/0语言进行分析，证明其可以进行自上而下的语法分析；
2. 对block、proc、statement、condition、expression、term、factor进行分析，画出语法分析图，在此基础上描述这些子程序的设计思想；
3. 具有一定的语法错误处理能力；

# 系统设计

## 系统结构



编译器的系统结构通常分为前端和后端2个主要部分。前端负责将源代码解析成中间表示，包括词法分析、语法分析和语义分析。此部分确保源代码的正确性并生成抽象语法树。接下来，后端会对中间表示进行各种优化，以提高代码效率。并将优化后的中间表示转换为目标代码，负责代码生成和机器指令的安排。整体上，这一结构帮助编译器高效地将高级语言转换为低级语言，实现程序的可执行性。