**编译原理实验1-词法分析**

**一、实验目的与任务**

通过设计编制调试一个具体的词法分析程序，加深对词法分析原理的理解。并掌握在对程序设计语言源程序进行扫描过程中将其分解为各类单词的词法分析方法。编制一个读单词过程，从输入的源程序中，识别出各个具有独立意义的单词，即基本保留字、标识符、常数、运算符、分隔符五大类，遇到错误时可显示“Error”，然后跳过错误部分继续显示，并依次输出各个单词的内部编码及单词符号自身值。

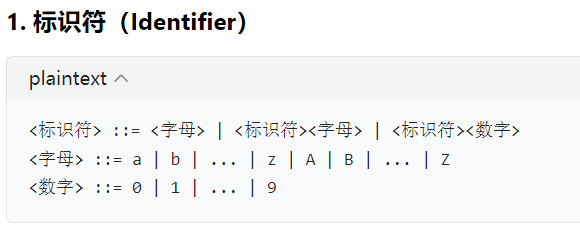
**二、实验涉及的相关知识点**

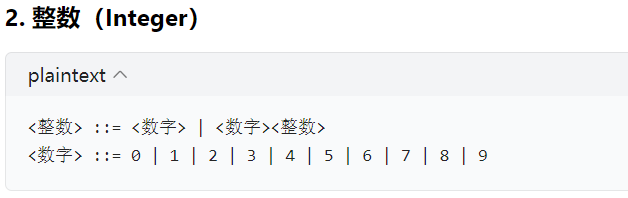
1. 词法分析器的功能和输出格式

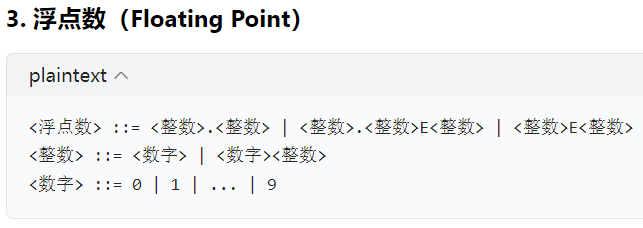
词法分析器的功能是输入源程序，输出单词符号。词法分析器的单词符号常常表示成以下的二元式(单词种别码，单词符号的属性值)。本实验中，采用的是一类符号一种别码的方式。

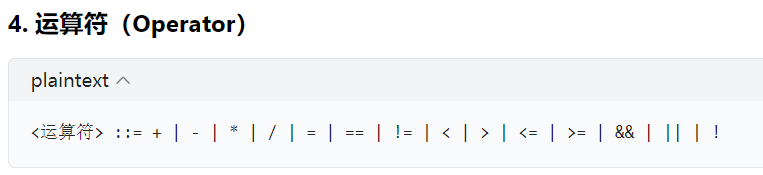
1. 单词的BNF表示

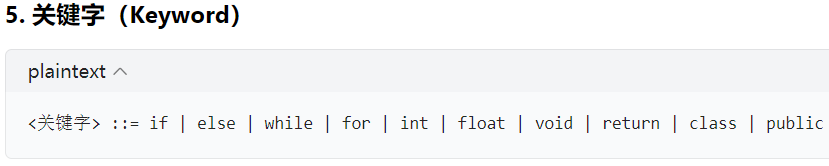
在编译原理中，BNF（巴科斯 - 诺尔范式）是一种用于精确描述编程语言语法的形式化表示方法。单词（Token）作为编译过程中的基本语法单位，可以用 BNF 进行严格定义。以下是编译原理中常见单词类型的 BNF 表示：













3.“超前搜索”方法

在编译原理的词法分析阶段，超前搜索（Lookahead） 是一种重要的技术，用于解决词法规则中的二义性和最大匹配原则问题。当词法分析器无法仅通过当前字符确定完整的词法单元（Token）时，需要向前查看若干个字符以做出正确判断。

**三、实验内容**

如源程序为C语言，代码如下文本框：

进行词法分析，要求如下：

main()

{ int a,b;

a = 10;

b = a + 20; }

|  |  |  |
| --- | --- | --- |
| 单词种别码 | 识别内容 | 示例 |
| 1 | 保留字 | if、int、for、while、do、return、break、continue |
| 2 | 标识符 | a,b等变量 |
| 3 | 无符号整数 | 整数常数 |
| 4 | 运算符 | +、-、\*、/、=、>、<、>=、<=、!= |
| 5 | 界符 | ,、;、{、}、(、) |

尝试编写代码，输出上述源代码的词法结果示例（圆圈序号不用展示）：

1. （2，”main”）
2. （5，”（“）
3. （5，”）“）
4. （5，”{“）
5. （1，”int”）
6. （2，”a”）
7. （5，”,”）
8. （2，”b”）
9. （5，”;”）
10. （2，”a”）
11. （4，”=”）
12. （3，”10”）
13. （5，”;”）
14. （2，”b”）
15. （4，”=”）
16. （2，”a”）
17. （4，”+”）
18. （3，”20”）
19. （5，”;”）
20. （5，”}“）
21. **实验步骤**
22. 通过生成式AI提示词工程，生成面向C语言的词法分析程序，并尝试逐步规范成本实验要求的实验内容。验证代码，分析AI生成的各函数模块的功能，并根据函数模块调用的顺序和过程，绘制词法分析处理流程图。
23. 验证参考代码，分析输出结果单词含义，并尝试验证如何新增关键字，并以你的姓名拼音为新增关键字进行验证，记录10条以上不同输入的实验结果。
24. **实验思考**
25. 为什么词法分析器无法识别语法错误（如括号不匹配、变量未声明）？
26. 举例说明哪些错误必须在语法分析阶段才能被检测到。
27. 如何实现 "超前扫描"（Lookahead）以处理歧义性（如 C＋＋语言中的>>操作符）？
28. 如何识别浮点数（如3.14或1.2e-5）？需要增加哪些词法规则？