词法分析

- 周闯
- 320170922001
- chzhou17@lzu.edu.cn

实验原理分析

词法分析
词法分析工作由词法分析器完成，又称扫描器。词法分析器输出为等长度的单词符号串
例如，给定如下输入：
position = initial + rate * 60
词法分析器将识别出 7 个单词符号
position, =, initial, +, rate, *, 60

算法流程分析

1.1

为了实现剔除注释的功能，程序需要将注释内的内容移除，需要注意的是C语言注释有两种情况：

1. 形如： //...的单行注释。对于这种注释不需要考虑到换行的问题
2. 形如： /.../ 的多行注释。对于这种注释需要考虑到换行问题

1.2

为了实现标识符的识别，首先需要了解构词规则：以字母打头，后跟任意多个字母、数字的单词；长度不超过15；不区分大小写；下划线视为第27个字母。

同时还需要区分关键字以及“ ”内的内容是不记录到标识符之中的，需要移除。

1.3

为了从任意字符串中识别所有合法的单词符号，需要借鉴1.2中的方法识别关键字，标识符；同时还需要对操作符，数字等等情况进行多维的分析，最后将其以二元组形式输出。

程序关键部分分析

1.1

定义：

flag：表示当前字符处于的位置； 0表示无注释；1表示//注释；2表示/*注释

str[]：原始字符串

str1[]：去掉注释后的字符串

关键部分分析：

此部分较难的部分在于对于/* */中换行规则的掌握，简单概括来说就是：如果开头结尾标志都在同一行的话则只需将其中的内同全部移除，不需要添加其他步骤。然而如果开头结尾标志不在同一行的话则需要判断结尾后面是否是换行符，如果后面不是换行符的话则需要自动添加一个换行符。

结果分析：

成功将两种注释中的内容合法的移除，并保持移除后内容的规范

1.2

定义：

is_key()：判断传入字符串是否为关键字

is_letter()：判断掺入的字符是否为字母或_

is_word()：判断传入字符串是否构词规则

find_id()：从传入的字符串中识别标识符

flag： find_id函数中的状态变量，flag=0：无特殊状态；flag=1：录入单词过程中；flag=2：处于引号内的内容；

str[]：源字符串

str1[]：录入单词暂存的位置

关键部分分析：

了解标识符的构词法，以字母打头，后跟任意多个字母、数字的单词；长度不超过15；不区分大小写；下划线视为第27个字母。然后根据构词规则设计算法实现从字符串中读出相应的标识符。在识别字符串的同时需要注意“ ”符号内部的字符串是不计入到标识符中的。

结果分析：

成功识别出所有不在“ ”内的标识符并打印

1.3

定义：

is_number()：判断传入字符是否为数字

is_operation()：判断传入字符是否为运算符

find_id_index()：查找标识符的序号

is_id()：判断传入的字符串是否为标识符

is_key()：判断传入字符串是否为关键字

is_letter()：判断掺入的字符是否为字母或_

is_word()：判断传入字符串是否构词规则

find_id()：从传入的字符串中识别标识符

flag： find_id函数中的状态变量，flag=0：无特殊状态；flag=1：录入单词过程中；flag=2：处于引号内的内容；flag=3：录入数字过程中

str[]：源字符串

str1[]：录入单词暂存的位置

关键部分分析：

1.3其实可以看做是1.2的拓展，该实验不仅要求识别标识符，还要求对关键字、运算符、常数进行区分并且对标识符进行保存。该实验的关键之处在于需要对于各个不同类型的单词种类的构词方法进行了解，并且通过代码实现出来。对于有穷集合关键字、运算符和无穷集合标识符、常数两种类型需要不同的处理方法。又穷集合可以通过数组来存储他们的所有内容；而无穷集合则需要利用链表来储存未知个

数的单词。

结果分析：

成功将所有单词通过二元组的方式实现出来。

实验总结

1. 编译环境：Code::Blocks 17.12
2. 语言环境：C++
3. 自我分析：这次实验是编译原理的第一次实验，也是最为基础的实验。由于初学编译原理，所以一开始对于编译原理实验的做法是有一定不了解的。但是，通过实验课上的摸索以及编译原理课程的推进，我自己对编译原理的理解也有了许多的加深，因此，许多之前看起来棘手的问题也逐渐有了可以解决的思路。同时，随着编译原理课程的不断推进，我也感受到了之后实验的难度会不断的上涨，而我们需要不断的提升对于编译原理的了解，并且进行深入的理解以便于将具体问题通过代码实现。