**江苏科技大学**

**实 验 报 告**

课 程： 编译原理

课 题： 实验1·词法分析设计

学 院： 计算机学院

学 号： 182210710119

姓 名： 陈四贵

班 级： 1822107101

指导老师： 肖 寒

# 一、实验目的

通过本实验的编程实践，了解词法分析的任务，掌握词法分析程序设计的原理和构造方法，以此加深对编译的基本概念、原理和方法有完整的和清楚的理解，并能正确地、熟练地运用。

# 二、设备与环境

windows 10环境、visual studio 2017

# 三、实验要求

用VC++/VB/JAVA语言实现对C语言子集的源程序进行词法分析。通过输入源程序从左到右对字符串进行扫描和分解，依次输出各个单词的内部编码及单词符号自身值；若遇到错误则显示“Error”，然后跳过错误部分继续显示 ；同时进行标识符登记符号表的管理。

# 四、实验原理

1、程序功能

输入：所给文法的源程序字符串

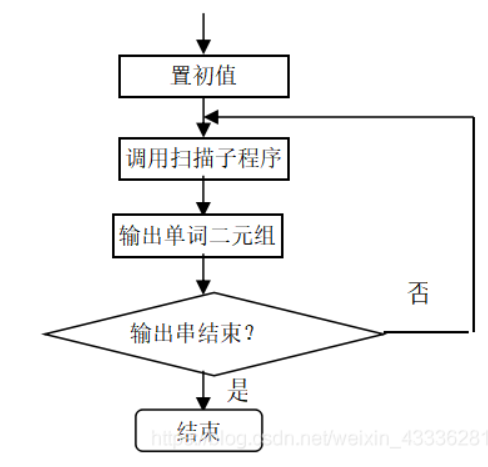
输出：二元组（syn,token或sum）构成的序列。

syn为单词种别码；

token为存放的单词自身字符串；

sum为整形常数。

例如：对源程序begin x:=9;if x>0 then x:=2\*x+1/3;end# 经词法分析后输出如下序列：（1，begin）(10,’x’) (18,:=) (11,9) (26,; ) (2,if)……

2、程序架构

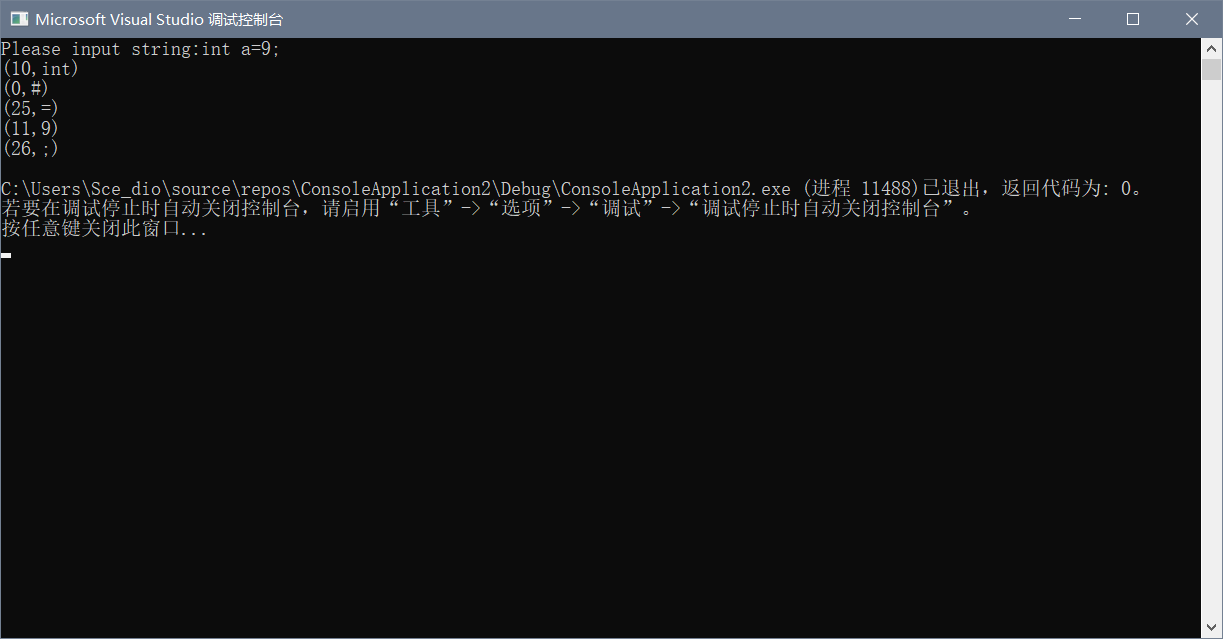
3、算法分析

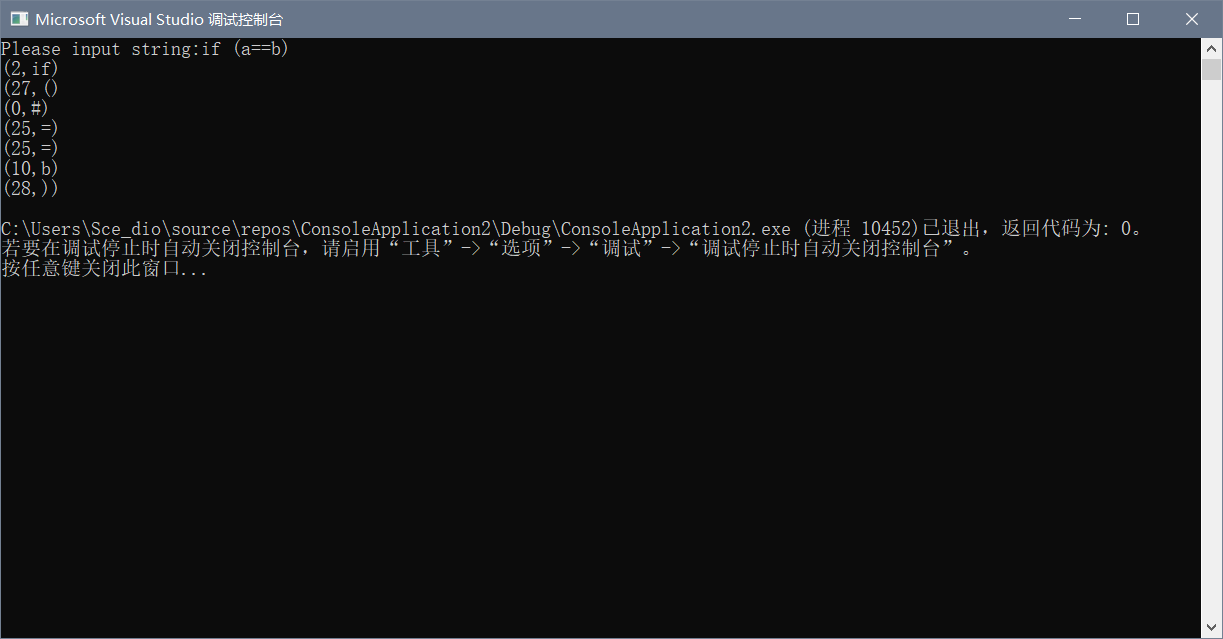
通过词法分析程序，依次读取用户输入的单词二元组，以首符号判断可能属于哪一类成分，即判断可能是单词、数字亦或是字符。对每一类成分识别子程序中，若截取到不符该成分组成规则的字符，应注意回退。

# 五、程序测试与调试

1、测试用例

1. //赋值语句int a=9;
2. //判断语句 if(a==b)

2、运行结果参考



# 六、程序源码

1. #include <iostream>
2. #include <string>
4. **using** **namespace** std;
6. // 关键字表置初始值，序号为类型编码，值为关键字。
7. string keyword[30] = { "#", "begin", "if", "then", "while", "do", "end", "", "", "",
8. "letter(letter|digit)\*", "digitdigit\*", "", "+", "-", "\*", "/",
9. ":", ":=", "", "<", "<>", "<=", ">", ">=", "=", ";", "(", ")" };
11. **class** word {//识别类，它的对象是一个识别出的单位，即是单词、还是关键字。
12. **public**:
13. **int** syn{};//类型编码
14. string token;//值
15. };

18. /\*以下这些函数都不是类里面包含的函数\*/
19. // 处理单词的函数
20. word letterAnalysis(**const** string &subCode) {
21. word item;
22. //看此单词是否是如下列出的“关键字”
23. **if** (subCode.substr(0, 5) == "begin") {
24. item.syn = 1;
25. }
26. **else** **if** (subCode.substr(0, 2) == "if") {
27. item.syn = 2;
28. }
29. **else** **if** (subCode.substr(0, 4) == "then") {
30. item.syn = 3;
31. }
32. **else** **if** (subCode.substr(0, 5) == "while") {
33. item.syn = 4;
34. }
35. **else** **if** (subCode.substr(0, 2) == "do") {
36. item.syn = 5;
37. }
38. **else** **if** (subCode.substr(0, 3) == "end") {
39. item.syn = 6;
40. }
41. **else** {
42. // 如果是其它单词，截取到第一个非字符
43. **for** (**int** i = 0; i < subCode.length(); ++i) {
44. **if** (!(subCode[i] > 'a' && subCode[i] < 'z')) {
45. item.syn = 10;
46. keyword[item.syn] = subCode.substr(0, i);
47. **break**;
48. }
49. }
50. }
51. item.token = keyword[item.syn];
52. **return** item;
53. }
55. // 处理数字的函数
56. word numberAnalysis(string subCode) {
57. word item;
58. item.syn = 11;
59. **for** (**int** i = 0; i < subCode.length(); ++i) {
60. // 截取到第一个非数字字符
61. **if** (!(subCode[i] >= '0' && subCode[i] <= '9')) {
62. keyword[item.syn] = subCode.substr(0, i);
63. **break**;
64. }
65. }
66. item.token = keyword[item.syn];
67. **return** item;
68. }
70. // 处理字符的函数
71. word charAnalysis(string subCode) {
72. word item;
73. **switch** (subCode[0]) {
74. **case** '#':
75. item.syn = 0;
76. **break**;
77. **case** '+':
78. item.syn = 13;
79. **break**;
80. **case** '-':
81. item.syn = 14;
82. **break**;
83. **case** '\*':
84. item.syn = 15;
85. **break**;
86. **case** '/':
87. item.syn = 16;
88. **break**;
89. **case** ':':
90. **if** (subCode[1] == '=') {//即识别出":="运算符
91. item.syn = 18;
92. }
93. **else** {//即识别出"//"运算符
94. item.syn = 17;
95. }
96. **break**;
97. **case** '<':
98. **if** (subCode[1] == '>') {//即识别出"<>"
99. item.syn = 21;
100. }
101. **else** **if** (subCode[1] == '=') {//即识别出"<="
102. item.syn = 22;
103. }
104. **else** {//即"<"
105. item.syn = 20;
106. }
107. **break**;
108. **case** '>':
109. **if** (subCode[1] == '=') {//即识别出">="
110. item.syn = 24;
111. }
112. **else** {//即识别出">"
113. item.syn = 23;
114. }
115. **break**;
116. **case** '=':
117. item.syn = 25;
118. **break**;
119. **case** ';':
120. item.syn = 26;
121. **break**;
122. **case** '(':
123. item.syn = 27;
124. **break**;
125. **case** ')':
126. item.syn = 28;
127. **break**;
128. }
129. item.token = keyword[item.syn];
130. **return** item;
131. }

134. **void** scanner(**const** string &code) {// 词法分析函数。参数为const string型，函数不改变字符串值，也改变不了。
135. //遍历读入的字符串
136. **for** (**int** i = 0; i < code.length(); ++i) {
137. word item;
138. **if** (code[i] > 'a' && code[i] < 'z') {
139. // 开头是字母，极可能是单词->处理单词
140. item = letterAnalysis(code.substr(i, code.length() - i + 1));
141. }
142. **else** **if** (code[i] >= '0' and code[i] <= '9') {
143. // 开头是数字，极可能是数字->处理数字
144. item = numberAnalysis(code.substr(i, code.length() - i + 1));
145. }
146. **else** **if** (code[i] == ' ') {
147. // 如果是空格，直接跳过
148. **continue**;
149. }
150. **else** {
151. // 处理特殊符号
152. item = charAnalysis(code.substr(i, code.length() - i + 1));
153. }
154. i += **int**(item.token.length()) - 1;
155. cout << "(" << item.syn << "," << item.token << ")" << endl;
156. }
157. }
159. **int** main() {
160. string code;
161. cout << "Please input string:";
162. // 读入一行代码，因为代码中有空格，所以要用 getline
163. getline(cin, code);
164. //调用scanner函数，分析读入的字符串code
165. scanner(code);
166. **return** 0;
167. }