中山大学数据科学与计算机学院本科生实验报告

课程名称：编译器构造实验 任课教师：陈炬桦

|  |  |  |  |
| --- | --- | --- | --- |
| 学年度 | 2018-2019 | 学期 | 第二学期 |
| 年级 | 16级 | 专业（方向） | 计算机科学与技术 |
| 学号 | 16337341 | 姓名 | 朱志儒 |
| 电话 | 15989184223 | Email | 739741104@qq.com |
| 开始日期 | 2019/3/29 | 完成日期 | 2019/3/29 |

1. 实验题目

**词法分析程序设计**

* 1. Description

设一语言的关键词、运算符、分界符的个数与单词如下：   
struct { int number; string str[10]; } keywords={3,"int","main","return"} ; //关键词  
struct { int number; string str[10]; } operators ={5,"+","\*","=","+=","\*="}; //运算符  
struct { int number; string str[10]; } boundaries ={6,"(",")","{","}",",",";"} ; //分界符  
struct { int number; string str[100];} identifieres={0}; //标识符  
struct { int number; string str[100];} Unsigned\_integer={0}; //无符号整数  
以上类号分别为1~5，序号从0开始；  
标识符是字母开头的字母数字串；常量为无符号整数；  
用C++设计一程序实现词法分析。

* 1. Input

输入一程序，结束符用”#”；

* 1. Output

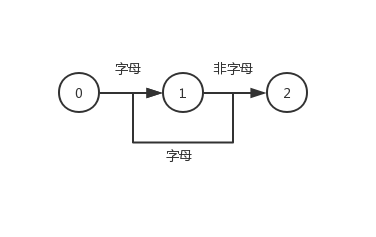
输出单词数对：<类号，序号>。 输出标识符表，用空格分隔； 输出无符号整数表，用空格分隔；

1. 算法描述（介绍程序模块功能，流程图）

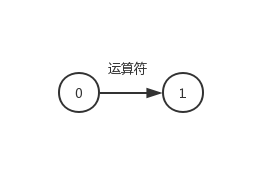
**预处理**：

删除无用的空格、跳格、回车和换行等编辑性字符，以及注释部分，再将处理后的字符装入一个缓冲区。

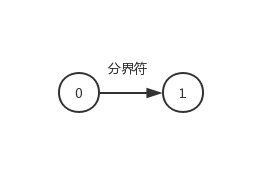
**关键词状态转换图：**



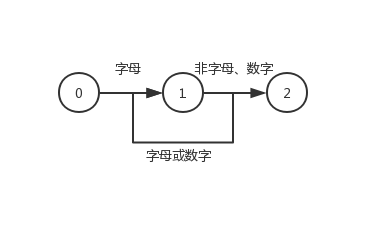
**运算符状态转换图：**



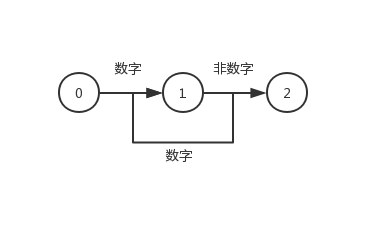
**分界符状态转换图：**



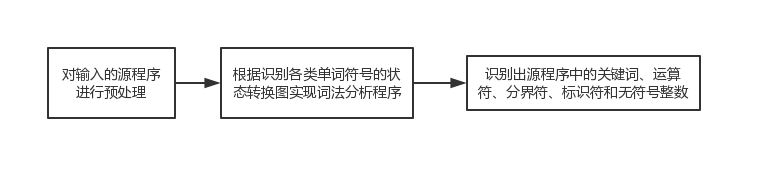
**标识符状态转换图：**



**无符号整数状态转换图：**



**流程图：**



1. 测试数据（2组）

|  |  |  |
| --- | --- | --- |
|  | 输入 | 输出 |
| 第1组 | main()  { int a=2,b=3;  return 2\*b+a;  }# | <1,1><3,0><3,1><3,2><1,0><4,0><2,2><5,0><3,4><4,1><2,2><5,1><3,5><1,2><5,0><2,1><4,1><2,0><4,0><3,5><3,3>  identifieres:a b  Unsigned\_integer:2 3 |
| 第2组 | int main() {  int happy1 = 100, happy2 = 200;  return (happy1 + 1) \* happy2;  } # | <1,0><1,1><3,0><3,1><3,2><1,0><4,0><2,2><5,0><3,4><4,1><2,2><5,1><3,5><1,2><3,0><4,0><2,0><5,2><3,1><2,1><4,1><3,5><3,3>  identifieres:happy1 happy2  Unsigned\_integer:100 200 1 |

1. 程序清单
2. #include <iostream>
3. #include <string>
4. **using** **namespace** std;
6. **struct** { **int** number; string str[10]; } keywords = { 3,"int","main","return" }; //关键词
7. **struct** { **int** number; string str[10]; } operators = { 5,"+","\*","=","+=","\*=" }; //运算符
8. **struct** { **int** number; string str[10]; } boundaries = { 6,"(",")","{","}",",",";" }; //分界符
9. **struct** { **int** number; string str[100]; } identifieres = { 0 }; //标识符
10. **struct** { **int** number; string str[100]; } Unsigned\_integer = { 0 }; //无符号整数
12. **bool** is\_letter(**char** in) {
13. **return** in >= 'a' && in <= 'z' || in >= 'A' && in <= 'Z';
14. }
16. **int** is\_operator(**char** in) {
17. **for** (**int** i = 0; i < operators.number; ++i)
18. **if** (in == operators.str[i][0])
19. **return** i;
20. **return** -1;
21. }
23. **int** is\_boundary(**char** in) {
24. **for** (**int** i = 0; i < boundaries.number; ++i)
25. **if** (in == boundaries.str[i][0])
26. **return** i;
27. **return** -1;
28. }
30. **bool** is\_number(**char** in) {
31. **return** in >= '0' && in <= '9';
32. }
34. **int** in\_identifieres(string in) {
35. **for** (**int** i = 0; i < identifieres.number; ++i)
36. **if** (in == identifieres.str[i])
37. **return** i;
38. **return** -1;
39. }
41. **int** is\_keyword(string in) {
42. **for** (**int** i = 0; i < keywords.number; ++i)
43. **if** (in == keywords.str[i])
44. **return** i;
45. **return** -1;
46. }
48. **int** in\_Ui(string in) {
49. **for** (**int** i = 0; i < Unsigned\_integer.number; ++i)
50. **if** (in == Unsigned\_integer.str[i])
51. **return** i;
52. **return** -1;
53. }
55. **int** main() {
56. string source;
57. **while** (**true**) {
58. string input;
59. getline(cin, input);
60. **if** (input.empty()) **break**;
61. source += input;
62. }
63. **int** i = 0;
64. **while** (i < source.size()) {
65. **if** (is\_letter(source[i])) {
66. string kword;
67. kword += source[i++];
68. **while** (is\_letter(source[i]) || is\_number(source[i]))
69. kword += source[i++];
70. **if** (is\_keyword(kword) != -1) {
71. cout << "<1," << is\_keyword(kword) << ">";
72. **continue**;
73. }
74. **if** (in\_identifieres(kword) != -1)
75. cout << "<4," << in\_identifieres(kword) << ">";
76. **else** {
77. identifieres.str[identifieres.number++] = kword;
78. cout << "<4," << in\_identifieres(kword) << ">";
79. }
80. **continue**;
81. }
82. **if** (is\_operator(source[i]) != -1) {
83. cout << "<2," << is\_operator(source[i++]) << ">";
84. **continue**;
85. }
86. **if** (is\_boundary(source[i]) != -1) {
87. cout << "<3," << is\_boundary(source[i++]) << ">";
88. **continue**;
89. }
90. **if** (is\_number(source[i])) {
91. string number;
92. number += source[i++];
93. **while** (is\_number(source[i]))
94. number += source[i++];
95. **if** (in\_Ui(number) != -1)
96. cout << "<5," << in\_Ui(number) << ">";
97. **else** {
98. Unsigned\_integer.str[Unsigned\_integer.number++] = number;
99. cout << "<5," << in\_Ui(number) << ">";
100. }
101. **continue**;
102. }
103. **if** (source[i] == '#') **break**;
104. i++;
105. }
106. cout << endl << "identifieres:";
107. **for** (**int** j = 0; j < identifieres.number; ++j)
108. cout << identifieres.str[j] << ' ';
109. cout << endl << "Unsigned\_integer:";
110. **for** (**int** j = 0; j < Unsigned\_integer.number; ++j)
111. cout << Unsigned\_integer.str[j] << ' ';
112. **return** 0;
113. }