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 教学助理（TA）：

|  |  |  |  |
| --- | --- | --- | --- |
| 学年度 | 2018-2019 | 学期 | 第二学期 |
| 年级 | 2016 | 专业（方向） | 计算机科学与技术 |
| 学号 | 16337052 | 姓名 | 杜尔鑫 |
| 电话 | 15626281204 | Email | duerx@mail2.sysu.edu.cn |
| 开始日期 | 2019-03-29 | 完成日期 | 2019-03-29 |

1. 实验题目

**1.1 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。

此题需要提交实验报告；“实验报告用“学号+姓名+51”

**1.2 Input**

输入一程序，结束符用”#”；

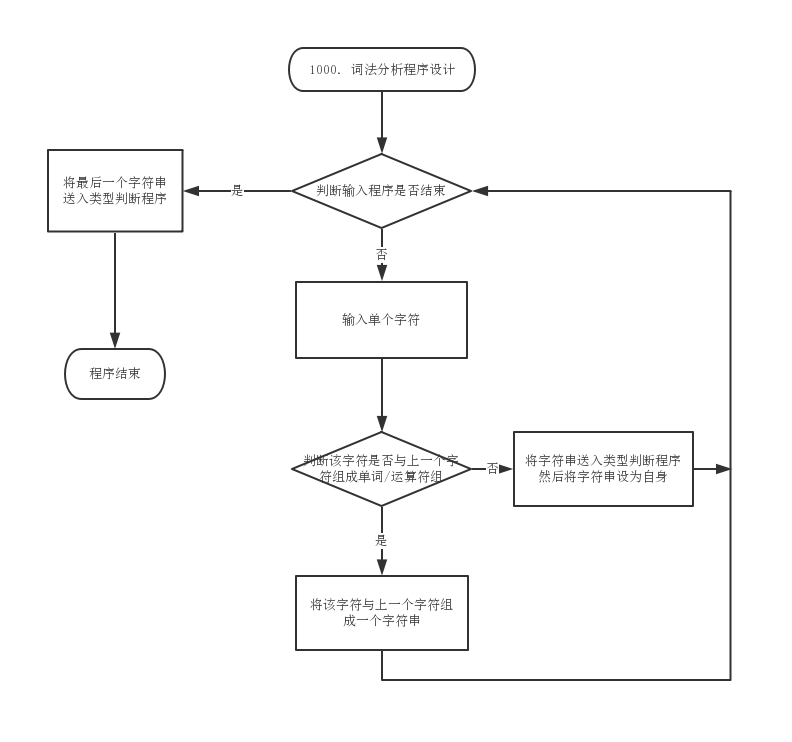
**1.3 Output**

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；

2. 算法描述(介绍程序模块功能；流程图)

1. 逐个输入字符，如果不为#，则进入2),否则进入3)；
2. 判断判断该字符是否与上一个字符串组成单词/运算符组，如果可以，组合成字符串，返回1)。否则将上一个字符串送入类型判断程，并将自己作为新的字符串；
3. 将最后一个字符串送入类型判断程序。

注：类型判断程序是识别字符串为关键词/运算符/分界符的程序，并识别序号。或者存储新变量和数字并分配序号。



3. 测试数据(2组)

**3.1 Standard Example**

**3.1.1 Input**

|  |
| --- |
| main()  { int a=2,b=3;  return 2\*b+a;  }# |

**3.1.2 Output**

|  |
| --- |
| <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 |

**3.2 Addition Test**

**3.2.1 Input**

|  |
| --- |
| main()  { int va=2,vb=3,vc=0;  vc += 2;  return 2\*vb+va;  }# |

**3.2.2 Output**

|  |
| --- |
| <1,1><3,0><3,1><3,2><1,0><4,0><2,2><5,0><3,4><4,1><2,2><5,1><3,4><4,2><2,2><5,2><3,5><4,2><2,3><5,0><3,5><1,2><5,0><2,1><4,1><2,0><4,0><3,5><3,3>  identifieres:va vb vc  Unsigned\_integer:2 3 0 |

4. 程序清单

|  |
| --- |
| #include <iostream>  using namespace std;  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  void Process(string inString, int inType)  {      for (int i = 0; i < keywords.number; ++i)      {          if (inString == keywords.str[i])          {              printf("<%d,%d>", 1, i);              return;          }      }      for (int i = 0; i < operators.number; ++i)      {          if (inString == operators.str[i])          {              printf("<%d,%d>", 2, i);              return;          }      }      for (int i = 0; i < boundaries.number; ++i)      {          if (inString == boundaries.str[i])          {              printf("<%d,%d>", 3, i);              return;          }      }      // undifined      if (inType == 4)      {          bool exitFlag = 0;          for (int i = 0; i < Unsigned\_integer.number; ++i)          {              if (inString == Unsigned\_integer.str[i])              {                  printf("<%d,%d>", 5, i);                  exitFlag = 1;                  return;              }          }          if (!exitFlag)          {              Unsigned\_integer.str[Unsigned\_integer.number] = inString;              printf("<%d,%d>", 5, Unsigned\_integer.number);              Unsigned\_integer.number ++;              return;          }      }      else      {          bool exitFlag = 0;          for (int i = 0; i < identifieres.number; ++i)          {              if (inString == identifieres.str[i])              {                  printf("<%d,%d>", 4, i);                  exitFlag = 1;                  return;              }          }          if (!exitFlag)          {              identifieres.str[identifieres.number] = inString;              printf("<%d,%d>", 4, identifieres.number);              identifieres.number ++;              return;          }      }  }  int main(int argc, char const \*argv[])  {      char inChar;      string lassString = "";      int thisType = 0, lassType = 0; // 0 blank, 1 letter, 2 operator, 3 bound, 4 integer      while (1)      {          inChar = getchar();          if (inChar == '#')          {              Process(lassString, lassType);              // return 0;              break;          }          if (inChar == '\n')              continue;          else if ((inChar >= 'a' && inChar <= 'z') || (inChar >= 'A' && inChar <= 'Z'))              thisType = 1;          else if (inChar == '+' || inChar == '\*' || inChar == '=')              thisType = 2;          else if (inChar >= '0' && inChar <= '9')              thisType = 4;          else if (inChar == ' ')              thisType = 0;          else              thisType = 3;          if (thisType == 3)          {              Process(lassString, lassType);              lassString = string(1, inChar);          }          else if (thisType == 0)          {              if (lassType != 0)                  Process(lassString, lassType);              lassString = "";          }          else if (lassType == 0 || thisType == lassType)          {              lassString = lassString + inChar;          }          else          {              Process(lassString, lassType);              lassString = string(1, inChar);          }          lassType = thisType;      }      cout << "\nidentifieres:";      for (int i = 0; i < identifieres.number; ++i)      {          cout << identifieres.str[i];          if (i != identifieres.number - 1)               cout << " ";      }      cout << "\nUnsigned\_integer:";      for (int i = 0; i < Unsigned\_integer.number; ++i)      {          cout << Unsigned\_integer.str[i];          if (i != Unsigned\_integer.number - 1)               cout << " ";      }      return 0;  } |