**-------- 2021 学年 第 1 学期**

**实验课程名称： 　　编译原理**

**开课学院： 　电子信息学部**

**指导老师姓名： 　段欣妤**

**学生姓名（学号）： 　陈文龙（**06**）**

**学生专业班级：**1982066

南航科技学院

学生实验报告

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **实验项目名称** | **实验1** | | | **成绩** |  |
| **姓名** | **陈文龙** | **专业班级** | 1982066 | **组别** |  |
| **同组者** |  | | | **日期** |  |
| 1. 实验目的   完成以下正则文法所描述的Pascal语言子集单词符号的词法分析程序。  <标识符>→字母︱ <标识符>字母︱ <标识符>数字  <无符号整数>→数字︱ <无符号整数>数字  <单字符分界符> →+ ︱－ ︱\* ︱; ︱(︱)  <双字符分界符>→<大于>=︱<小于>=︱<小于>>︱<冒号>=︱<斜竖>\*  <小于>→<  <等于>→=  <大于>→>  <冒号> →:  <斜竖> →/  该语言的保留字 ：void、int、float、double、if、else、for、do、while 等  说明： 1 该语言大小写不敏感。  2 字母为a-z A-Z，数字为0-9。  3 可以对上述文法进行扩充和改造。  4‘/\*……\*/’为程序的注释部分。   1. 实验内容   1、给出各单词符号的类别编码。  2、词法分析程序应能发现输入串中的错误。  3、词法分析作为单独一遍编写，词法分析结果为二元式序列组成的中间文件。  4、设计两个测试用例（尽可能完备），并给出测试结果。  三、实验步骤  1、文法扩充修改  （1）将每个单字符作为一种类别进行分析；  （2）加入了保留字 main；  （3）扩充双字符，并将每一个种类的双字符作为一种类别；  （4）加入左右注释和单行注释。  2、程序功能描述  （1）能够识别.c文件中的标识符、无符号整数；  （2）能够识别.c文件中的部分关键字：void、int、float、double、if、else、  for、do、while、main（等等）；  （3）能够识别一个.c文件中的算术运算符和逻辑运算符等字符符号；  （4）词法分析结果为二元式组成的中间文件。  3、流程图    4、程序结构  （1）词法分析程序LexicalAnalysisProcess()的主体是一个不断扫描输入串的扫描器和配合该扫描器进行分析的符号表。描述符号表的类SymbolTable和描述扫描器动作的类LexicalScanner。  （2）符号表symbolTableList  符号表symbolTableList采用List<String>列表保存，在程序中写入，而各类单词符号的类别编码正好可以用其在列表中的序号表示。  private static List<String> symbolTableList = new ArrayList<String>();  符号表内容：  symbolTableList.add("标识符");  symbolTableList.add("无符号整数");  symbolTableList.add("单字符分界符");  symbolTableList.add("void");  symbolTableList.add("int");  symbolTableList.add("float");  symbolTableList.add("double");  symbolTableList.add("if");  symbolTableList.add("else");  symbolTableList.add("for");  symbolTableList.add("do");  symbolTableList.add("while");  symbolTableList.add("main");  symbolTableList.add("加号");  symbolTableList.add("减号");  symbolTableList.add("位与");  symbolTableList.add("位或");  symbolTableList.add("大于");  symbolTableList.add("小于");  symbolTableList.add("星");  symbolTableList.add("斜竖");  symbolTableList.add("感叹号");  symbolTableList.add("赋值");  symbolTableList.add("大于等于");  symbolTableList.add("右移");  symbolTableList.add("小于等于");  symbolTableList.add("左移");  symbolTableList.add("不等于");  symbolTableList.add("等于");  symbolTableList.add("左注释");  symbolTableList.add("右注释");  symbolTableList.add("单行注释");  symbolTableList.add("加等于");  symbolTableList.add("减等于");  symbolTableList.add("乘等于");  symbolTableList.add("除等于");  symbolTableList.add("与等于");  symbolTableList.add("或等于");  symbolTableList.add("加加");  symbolTableList.add("减减");  symbolTableList.add("与");  symbolTableList.add("或");  四、源代码   1. Lexicalanalysis.java   import java.io.File;  import java.io.FileNotFoundException;  import java.io.FileReader;  import java.io.FileWriter;  import java.io.IOException;  import java.io.PushbackReader;  import java.util.ArrayList;  import java.util.List;  public class Lexicalanalysis {      public static void main(String[] args) {          //1：构造识别文本的符号表          SymbolTable st = SymbolTable.CreateSymbolTable();          //2：构造扫描器          LexicalScanner sc = new LexicalScanner("test1.c");          //3:词法分析          sc.LexicalAnalysisProcess();      }  }  /\*\*处理输入流的扫描器类   \* 私有成员变量：String taken —— 待识别的字符串或字符   \* 私有成员变量：String value —— 保存到符号表的识别成功的字符串或字符   \* 私有函数 lookupSymbolTableList() 以taken查询字符串保留字表，返回相应的关键字类别码，否则返回0   \* 私有函数 outMidFile() 进入终态输出二元式   \* 词法分析主要程序LexicalAnalysisProcess()   \* \*/   class LexicalScanner {      private String taken = null;      private String value = null;      private int categoryCode = 0;      private FileReader fr = null;      private FileWriter fw = null;      private PushbackReader pr = null;      private boolean midFileCreateSuccess = true;//是否有错误标志，如果有错误则不创建中间文件      private String outMidFileName = "";      //构造函数，输入是文件名      public LexicalScanner(String fileName) {          File fp = new File(fileName);            //检查文件后缀名是否合法          if(!fp.getName().endsWith(".c")) {              System.out.println("文件格式不正确...");              return;          }            //构造文件阅读器          try {              fr = new FileReader(fp);              pr = new PushbackReader(fr,2); //用可以回退的回退流封装，并设置两个字符大小的缓存          } catch (FileNotFoundException e) {              // TODO Auto-generated catch block              System.out.println(fileName+"文件不存在...");              e.printStackTrace();          }            // 构造输出文件          outMidFileName = fileName.substring(0, fileName.indexOf('.')) + ".tys";          fp = new File(outMidFileName);          try {              if (!fp.exists()) {                  fp.createNewFile(); // 创建输出的中间文件              }              fw = new FileWriter(fp);          } catch (IOException e) {              // TODO Auto-generated catch block              e.printStackTrace();          }      }        //词法分析      public void LexicalAnalysisProcess() {          if(pr == null) {              System.out.println("文件错误...");              return;          }            char ch;          int state;          boolean keepScan = true;          while(true) {              boolean error = false;              try {                  ch = (char) pr.read();                  if(ch==(char)-1) { //读取文件尾，说明文件分析完成，退出                      break;                  }                    switch(ch) {                  //单字符分界符                  case ';':case ',':case '(':case ')':case '{':case '}':                      taken = "单字符分界符";                      value = ""+ch;                      break;                    //+簇                  case '+':                      ch = (char)pr.read();                      switch(ch) {                      case '+':                          taken = "加加";                          value = "++";                          break;                      case '=':                          taken = "加等于";                          value = "+=";                          break;                      default:                          taken = "加号";                          value = "+";                          //指针回退                          pr.unread(ch);                          break;                      }                      break;                    //-簇                  case '-':                      ch = (char)pr.read();                      switch(ch) {                      case '-':                          taken = "减减";                          value = "--";                          break;                      case '=':                          taken = "减等于";                          value = "-=";                          break;                      default:                          taken = "减号";                          value = "-";                          pr.unread(ch);                          break;                      }                      break;                    //\*簇                  case '\*':                      ch = (char)pr.read();                      switch(ch) {                      case '=':                          taken = "乘等于";                          value = "\*=";                          break;                      case '/':                          taken = "右注释";                          value = "\*/";                          break;                      default:                          taken = "星";                          value = "\*";                          pr.unread(ch);                          break;                      }                      break;                    ///簇                  case '/':                      ch = (char)pr.read();                      switch(ch) {                      case '=':                          taken = "除等于";                          value = "/=";                          break;                      case '\*':                          taken = "左注释";                          value = "/\*";                            while(true)                          {                              ch = (char)pr.read();                              if(ch=='\*')                              {                                  ch = (char)pr.read();                                  if(ch=='/')                                  {                                      char buf[]=new char[2];                                      buf[0]='\*';                                      buf[1]='/';                                      pr.unread(buf);                                        break;                                  }                              }                          }                          break;                      case '/':                          taken = "单行注释";                          value = "//";                          while(true)                          {                              ch = (char)pr.read();                              if(ch=='\n')                              {                                  //读到\n表示开始读回车符，读取下一个\n;                                  pr.read();                                  break;                              }                          }                          break;                      default:                          taken = "斜竖";                          value = "/";                          pr.unread(ch);                          break;                      }                      break;                    //&簇                  case '&':                      ch = (char)pr.read();                      switch(ch) {                      case '=':                          taken = "与等于";                          value = "&=";                          break;                      case '&':                          taken = "与";                          value = "&&";                          break;                      default:                          taken = "位与";                          value = "&";                          pr.unread(ch);                          break;                      }                      break;                    //|簇                  case '|':                      ch = (char)pr.read();                      switch(ch) {                      case '=':                          taken = "或等于";                          value = "|=";                          break;                      case '|':                          taken = "或";                          value = "||";                          break;                      default:                          taken = "位或";                          value = "|";                          pr.unread(ch);                          break;                      }                      break;                  //=簇                  case '=':                      ch = (char)pr.read();                      switch(ch) {                      case '=':                          taken = "等于";                          value = "==";                          break;                      default:                          taken = "赋值";                          value = "=";                          pr.unread(ch);                          break;                        }                      break;                  //!簇                  case '!':                      ch = (char)pr.read();                      switch(ch) {                      case '=':                          taken = "不等于";                          value = "!=";                          break;                      default:                          taken = "感叹号";                          value = "!";                          pr.unread(ch);                          break;                      }                      break;                    //<簇                  case '<':                      ch = (char)pr.read();                      switch(ch) {                      case '=':                          taken = "小于等于";                          value = "<=";             break;                      case '<':                          taken = "左移";                          value = "<<";                          break;                      default:                          taken = "小于";                          value = "<";                          pr.unread(ch);                          break;                      }                      break;              //>簇                  case '>':                      ch = (char)pr.read();                      switch(ch) {                      case '=':                          taken = "大于等于";                          value = ">=";                          break;                      case '>':                          taken = "右移";                          value = ">>";                          break;                      default:                          taken = "大于";                          value = ">";                          pr.unread(ch);                          break;                      }                      break;                  //空格                  case ' ':case '\t':                      //不作为                      taken = null;                      break;                  case '\n':                      //读到\n表示开始读回车符，读取下一个\n;                      pr.read();                      taken = null;                      break;                  //标识符和整数簇                  default:                      //自成一个词法分析程序                      taken = "";                      value = null;                      keepScan = true;                      state = 0;                      while (keepScan) {                          switch(state) {                          case 0:                              if(!Character.isLetterOrDigit(ch)) {                                  //出错                                  keepScan = false;                                  break;                              }                              else if(ch == 'v')                                  state = 1;                              else if(ch == 'i')                                  state = 5;                              else if(ch == 'f')                                  state = 9;                              else if(ch == 'd')                                  state = 16;                              else if(ch == 'e')                                  state = 22;                              else if(ch == 'w')                                  state = 26;                              else if(ch == 'm')                                  state = 31;                              else if(ch >= '0' && ch <= '9')                                  state = 35;                              else                                  state = 36;                              break;                          case 1:                              if(ch == 'o')                                  state = 2;                              else{                                  state = 36;                                  pr.unread(ch);                                  ch = ' ';                              }                              break;                          case 2:                              if(ch == 'i')                                  state = 3;                              else {                                  state = 36;                                  pr.unread(ch);                                  ch = ' ';                              }                              break;                          case 3:                              if(ch == 'd')                                  state = 4;                              else {                                  state = 36;                                  pr.unread(ch);                                  ch = ' ';                              }                              break;                          case 4:                              //既不是字母也不是数字，说明识别出来了                              if(!Character.isLetterOrDigit(ch)) {                                  keepScan = false;                              }else {                                  state = 36;                              }                              pr.unread(ch);                              ch = ' ';                              break;                          case 5:                              if(ch == 'f')                                  state = 6;                              else if(ch == 'n')                                  state = 7;                              else {                                  state = 36;                                  pr.unread(ch);                                  ch = ' ';                              }                              break;                          case 6:                              if(!Character.isLetterOrDigit(ch)) {                                  keepScan = false;                              }                              else {                                  state = 36;                              }                              pr.unread(ch);                              ch = ' ';                              break;                          case 7:                              if(ch == 't')                                  state = 8;                              else {                                  state = 36;                                  pr.unread(ch);                                  ch = ' ';                              }                              break;                          case 8:                              if(!Character.isLetterOrDigit(ch)) {                                  keepScan = false;                              }                              else {                                  state = 36;                              }                              pr.unread(ch);                              ch = ' ';                              break;                          case 9:                              if(ch == 'l')                                  state = 10;                              else if(ch == 'o')                                  state = 14;                              else {                                  state = 36;                                  pr.unread(ch);                                  ch = ' ';                              }                              break;                          case 10:                              if(ch == 'o')                                  state = 11;                              else {                                  state = 36;                                  pr.unread(ch);                                  ch = ' ';                              }                              break;                          case 11:                              if(ch == 'a')                                  state = 12;                              else {                                  state = 36;                                  pr.unread(ch);                                  ch = ' ';                              }                              break;                          case 12:                              if(ch == 't')                                  state = 11;                              else {                                  state = 36;                                  pr.unread(ch);                                  ch = ' ';                              }                              break;                          case 13:                              if(!Character.isLetterOrDigit(ch)) {                                  keepScan = false;                              }                              else {                                  state = 36;                              }                              pr.unread(ch);                              ch = ' ';                              break;                          case 14:                              if(ch == 'r')                                  state = 15;                              else {                                  state = 36;                                  pr.unread(ch);                                  ch = ' ';                              }                              break;                          case 15:                              if(!Character.isLetterOrDigit(ch)) {                                  keepScan = false;                              }                              else {                                  state = 36;                              }                              pr.unread(ch);                              ch = ' ';                              break;                          case 16:                              if(ch == 'o')                                  state = 17;                              else {                                  state = 36;                                  pr.unread(ch);                                  ch = ' ';                              }                              break;                          case 17:                              if(!Character.isLetterOrDigit(ch)) {                                  keepScan = false;                                  pr.unread(ch);                                  ch = ' ';                              }                              else if(ch=='a'){                                  state = 18;                              }else {                                  state = 36;                                  pr.unread(ch);                                  ch = ' ';                              }                              break;                          case 18:                              if(ch == 'b')                                  state = 19;                              else {                                  state = 36;                                  pr.unread(ch);                                  ch = ' ';                              }                              break;                          case 19:                              if(ch == 'c')                                  state = 20;                              else {                                  state = 36;                                  pr.unread(ch);                                  ch = ' ';                              }                              break;                          case 20:                              if(ch == 'e')                                  state = 21;                              else {                                  state = 36;                                  pr.unread(ch);                                  ch = ' ';                              }                              break;                          case 21:                              if(!Character.isLetterOrDigit(ch)) {                                  keepScan = false;                              }                              else {                                  state = 36;                              }                              pr.unread(ch);                              ch = ' ';                              break;                          case 22:                              if(ch == 'l')                                  state = 23;                              else {                                  state = 36;                                  pr.unread(ch);                                  ch = ' ';                              }                              break;                          case 23:                              if(ch == 's')                                  state = 24;                              else {                                  state = 36;                                  pr.unread(ch);                                  ch = ' ';                              }                              break;                          case 24:                              if(ch == 'e')                                  state = 25;                              else {                                  state = 36;                                  pr.unread(ch);                                  ch = ' ';                              }                              break;                          case 25:                              if(!Character.isLetterOrDigit(ch)) {                                  keepScan = false;                              }                              else {                                  state = 36;                              }                              pr.unread(ch);                              ch = ' ';                              break;                          case 26:                              if(ch == 'h')                                  state = 27;                              else {                                  state = 36;                                  pr.unread(ch);                                  ch = ' ';                              }                              break;                          case 27:                              if(ch == 'i')                                  state = 28;                              else {                                  state = 36;                                  pr.unread(ch);                                  ch = ' ';                              }                              break;                          case 28:                              if(ch == 'l')                                  state = 29;                              else {                                  state = 36;                                  pr.unread(ch);                                  ch = ' ';                              }                              break;                          case 29:                              if(ch == 'e')                                  state = 30;                              else {                                  state = 36;                                  pr.unread(ch);                                  ch = ' ';                              }                              break;                          case 30:                              if(!Character.isLetterOrDigit(ch)) {                                  keepScan = false;                              }                              else {                                  state = 36;                              }                              pr.unread(ch);                              ch = ' ';                              break;                          case 31:                              if(ch == 'a')                                  state = 32;                              else {                                  state = 36;                                  pr.unread(ch);                                  ch = ' ';                              }                              break;                          case 32:                              if(ch == 'i')                                  state = 33;                              else {                                  state = 36;                                  pr.unread(ch);                                  ch = ' ';                              }                              break;                          case 33:                              if(ch == 'n')                                  state = 34;                              else {                                  state = 36;                                  pr.unread(ch);                                  ch = ' ';                              }                              break;                          case 34:                              if(!Character.isLetterOrDigit(ch)) {                                  keepScan = false;                              }                              else {                                  state = 36;                              }                              pr.unread(ch);                              ch = ' ';                              break;                          case 35:                              if(Character.isDigit(ch))                                  state = 35;                              else if(Character.isLetter(ch)) {                                  state = 35;                                  error = true;                              }                              else {                                  keepScan = false;                                  pr.unread(ch);                                  value = taken;                                  if(error) {                                      taken = "error";                                  }else {                                      taken = "无符号整数";                                  }                                  ch = ' ';                              }                              break;                          case 36:                              if(Character.isLetterOrDigit(ch))                                  state = 36  ;                              else {                                  keepScan = false;                                  pr.unread(ch);                                  value = taken;                                  taken = "标识符";                                  ch = ' ';                              }                              break;                          }                          if(ch!=' ')                              taken += ch;                          if(keepScan)                              ch = (char)pr.read(); //读取下一个字符                      }                      if(value==null)                          value = taken;                  break;                  }              } catch (IOException e) {                  // TODO Auto-generated catch block                  e.printStackTrace();              }                if(taken == null)                  continue;              //查表，输出              categoryCode = lookupSymbolTableList();              outMidFile(); //输出              taken = null;          }          //关闭文件          try {              fw.close();              if(!midFileCreateSuccess) {                  System.out.println("词法分析失败，二元式文件生成失败...");              }              else {                  System.out.println("词法分析完成，二元式文件生成成功...");              }          } catch (IOException e) {              // TODO Auto-generated catch block              e.printStackTrace();          }      }      private int lookupSymbolTableList() {          if(taken == null) {              return 0;          }          //返回类别号          return SymbolTable.getSymbolTableList().indexOf(taken)+1;      }        //输出二元式,向中间文件写入二元式      private void outMidFile() {          if(categoryCode==0) {              System.out.println("Error:未定义的标识符："+value);              this.midFileCreateSuccess = false;              return;          }          System.out.println("["+categoryCode+","+value+"]");          try {              fw.write("["+categoryCode+","+value+"]"+"\n");          } catch (IOException e) {              // TODO Auto-generated catch block              e.printStackTrace();          }      }      public boolean isCreateSuccess() {          return midFileCreateSuccess;      }      public String getMidFileName() {          return this.outMidFileName;      }  }   /\*\*配合词法分析的符号表    \* 由于全只有一个符号表，采用单例模式设计    \* 用java的集合类 list存放符号    \* \*/  class SymbolTable {      private static SymbolTable symbolTable; //本类      private static List<String> symbolTableList = new ArrayList<String>();        private SymbolTable(){}       public static synchronized SymbolTable CreateSymbolTable(){           if(symbolTable==null){              symbolTable = new SymbolTable();              //构造符号表              symbolTableList.add("标识符");              symbolTableList.add("无符号整数");              symbolTableList.add("单字符分界符");              symbolTableList.add("void");              symbolTableList.add("int");              symbolTableList.add("float");              symbolTableList.add("double");              symbolTableList.add("if");              symbolTableList.add("else");              symbolTableList.add("for");              symbolTableList.add("do");              symbolTableList.add("while");  symbolTableList.add("main");              symbolTableList.add("加号");              symbolTableList.add("减号");              symbolTableList.add("位与");              symbolTableList.add("位或");              symbolTableList.add("大于");              symbolTableList.add("小于");              symbolTableList.add("星");              symbolTableList.add("斜竖");              symbolTableList.add("感叹号");              symbolTableList.add("赋值");              symbolTableList.add("大于等于");              symbolTableList.add("右移");              symbolTableList.add("小于等于");              symbolTableList.add("左移");              symbolTableList.add("不等于");              symbolTableList.add("等于");              symbolTableList.add("左注释");              symbolTableList.add("右注释");              symbolTableList.add("单行注释");              symbolTableList.add("加等于");              symbolTableList.add("减等于");              symbolTableList.add("乘等于");              symbolTableList.add("除等于");              symbolTableList.add("与等于");              symbolTableList.add("或等于");              symbolTableList.add("加加");              symbolTableList.add("减减");              symbolTableList.add("与");              symbolTableList.add("或");           }           return symbolTable;       }       public static synchronized List<String> getSymbolTableList(){          return symbolTableList;       }   }   1. test1.c（测试案例1）   /\*\*左右注释   \* \*/  void main(){      int num1 = 123;      int num2 = 123;      float num3 = 1234;      double num4 = 12345;      for(int i = 0;i < 100;++i){          do{              if(num1==num2||num3!=num4){                  --num2;                  return 0;              }else{                  return num1&=num2;              }          }while(num1&&num2)      }      //单行注释      int d = num1<<5;      int c = num2>>6;      int a = num1+num2;      int b = num1-num2;      int c = num1\*num2;      int d = num1/num2;      a+=b;      a-=b;      a\*=b;      a/=b;      a|=b;      a&=b;  }   1. test2.c（测试案例2）   int a = 345;  int c = a%3;  int *2num* = 34;  五、测试结果   1. test1.c（测试成功）   词法分析结果中间文件：           1. test2.c（测试失败）   测试分析结果中间文件    六、实验总结  刚开始写实验的时候确实无从下手，后面通过看书和查阅资料了解大概，然后开始尝试写步骤、流程和程序，渐入佳境。通过这次实验让我对词法分析的理解更深了一步，也让我进一步地熟悉了java语言，提高了我的编程思维能力，增大了我的兴趣爱好，巩固了我的知识这也是做实验和学习这门课程的目的所在。一步步完善实验的过程也是在增强自己的知识。 | | | | | |