编译原理课程词法分析实验报告

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **专业班级** | | **计算机科学与技术5班** | **姓名** | | **李林轩** | **学号** | **20184320** |
| **实验题目** | 词法分析程序的设计与实现 | | | | | | |
| **实验时间** | **2021/4/10** | | | **实验地点** | **DS1421** | | |
| **实验成绩** |  | | | **实验性质** | **□验证性 □设计性 ■综合性** | | |
| 教师评价：  □算法/实验过程正确；□源程序/实验内容提交 □程序结构/实验步骤合理；  □实验结果正确； □语法、语义正确； □报告规范；  其他：  评价教师签名： | | | | | | | |
| 一、实验目的  🡪了解编译器中词法分析环节的作用和意义  🡪掌握词法分析基础  🡪实现实验要求中C语言子集的词法分析器程序 | | | | | | | |
| 二、实验项目内容   1. 实验内容   根据某个现有编程语言的词法规则（或设计一套新的词法规则），建立词法分析器展开识别，具体功能要求包括：   1. 支持标识符、常量（含十进制与十六进制的整数、浮点数）、关键字、分界符、运算符等词法类型； 2. 输出每个词的类型和单词属性； 3. 检查源程序中存在的词法错误，并报告错误所在的位置； 4. 查填符号表，例如，对于变量标识符，需在符号表存入标识符名字、类型等信息。 5. 实验要求： 6. 自选程序设计语言（java，python，c/c++）作为实现语言，手工编写词法分析程序。 7. 提交实验报告及源代码。实验报告需严格遵循学校文档规范，内容包含对应文法、词类编码表、词法分析测试用例。 | | | | | | | |
| 三、实验过程或算法 | | | | | | | |
| 内容要点，供参考：   1. 程序实现内容介绍 2. 语言说明：词法分析器所包含C语言子集相应的文法/正规式/状态转换图 3. 词法编码表及说明 4. 符号表/符号表接口及说明 5. 错误处理说明 6. 程序说明：可选择其中重点对象进行说明，不需要在这里列出全部源代码   一、程序实现内容介绍  本程序实现了以C语言的词法规则子集为基础，在此基础上进行一些修改而成的词法分析器。该词法分析器可识别标识符、常量（含十进制与十六进制的整数、浮点数）、关键字、分界符、运算符等词法类型，可以进行分词并输出每个词的类型与单词属性，可以检查源程序中存在的词法错误，并报告错误所在的行数与出错的单词，并且将输出自定义的错误类型。  二、语言说明  源程序中的记号，表达式，C语言子集文法：   1. 标识符：按照C语言的规则，标识符的正则表达式应为 [a-zA-z\_][a-zA-Z\_0-9]+ 2. 运算符：本词法分析器可识别的运算符包括：   +、-、\*、/、%、+=、-=、\*=、/=、==、!=、>、<、<=、>=、&、&&、|、||   1. 关键字：本词法分析器可识别的关键字是C语言的子集，可能有少量关键字不包括在内。关键字的单词属性用数字表示，一共分为了：   声明变量类： （int,float,struct等）  限定修饰符类： （const,unsigned,static等）  循环分支语句类：（for,while,break等）  输入输出类： （scanf,printf）  函数特有类： （void,return,volatile）  关键字及关键字对应的单词属性存储在vector<pair<string, int>> keyword   1. 数字：数字分为十进制整数，十六进制整数，十进制浮点数，十六进制浮点数四种，会分别进行类别的识别与输出。同时，词法分析器允许以小数点开头的浮点数，视为整数部分为0。 2. 分界符：分界符包括(、)、{、}、[、]、#、,、;、’、”   状态转换图：   1. 词法编码表及说明 2. 符号表/符号表接口及说明 3. 错误处理说明   1. 错误输出：  错误发生的行号将输出在terminal，错误发生的具体单词与错误类型将输出在result.txt文件中。  2. 错误定义：  1 数字错误：  当一串字符串以数字开头时，将默认其为数字，此时如果后面的字符不全为数字，则判定为数字错误。当然也有可能存在是以数字开头而造成的标识符命名错误，在本词法分析器中统一按照数字错误处理  2 浮点数字错误  当数字中出现了小数点时，状态则转为浮点数，浮点数出现错误与数字错误相似  七、程序说明  1. 程序输入形式：以文件输入  2. 程序输出形式：将词法分析的结果输出到文件中，末尾会输出对该程序的词法分析统计结果，包括每种词法的数量，字符总数，行数等；在屏幕上将输出错误，包括错误类型，错误所在行数，错误的单词。  3. 程序执行流程：  对文件内容逐行输入，对每一行逐个字符进行分析。根据当前读入字符进行状态转移，在实现的过程中没有明显的将状态表现出来（比如case xxx：），而是隐式的判断后进行状态转换。  变量和函数： | | | | | | | |
| 四、实验测试结果   1. 测试程序一   # include<stdio.h> int main(){  int num1,num2,op,ans;  scanf("%d%d%d",num1,num2,op);  if(op==0)  {  ans = num1 + num2;  ans ++;  };  if(op==1)  {  ans = num1 - num2;  ans --;  };  if(op==2)  {  ans = num1 & num2;  ans %= 10;  };  if(op==3)  {  ans = num1 | num2;  ans |= 10;  ans &= 15;  ans = ans\*\*2;  };   double f = 0.001;  double f\_err = 0.0.1;  double 9\_errID = 0x12.34abc;   printf("result = %d",ans);  }  控制台输出：  IMG_256  Result.txt文件内容（由于文件内容较长，此处放部分截图）  IMG_256  对于 |=，&=这类错误的运算符，该词法分析器可以进行识别并将错误类型输出在result.txt中，对于\*\*等双字符的运算符也能够正确识别  IMG_256  对于0.0.1这种不合法的浮点数与9\_errID这种不合法的标识符/数字，能够进行错误类型以及出错单词的输出  IMG_256  在result.txt文件末尾将输出对于词法分析测试程序的统计结果，包括程序所有单词，字符以及错误的数量，以及对于每个单词的数量统计。  测试结果分析  该测试测试了自定义的三种错误类型，在需要被词法分析器分析的程序里出现时能否正确的被识别以及输出。   1. 测试程序二   #include <stdio.h> int main() {  printf("Hello World");  return 0; }  控制台输出  IMG_256  当没有错误存在时，会输出Perfect! | | | | | | | |
| 五、实验总结  1、实验过程中所遇到的问题及解决办法  目前该词法分析器还有能够优化的部分，比如说在读入文件中\n，\t等代表的字符时可以正常运行，但是遇到’\n’，’\t’等在字符串中的形式时，会因为找不到下一个状态而进入死循环，后续可以引入输入\的新状态，将这部分进行优化。 | | | | | | | |