|  |  |  |
| --- | --- | --- |
| **得分** | **教师签名** | **批改日期** |
|  |  |  |

课程编号： IB00064



**深圳技术大学实验报告**

**课程名称： 编译原理**

**实验名称： 词法分析程序设计**

**班 级： 计算机科学与技术2班**

**指导教师： 袁明川、赵鹏**

**报 告 人： 黄荣权 学号： 202002020213**

**合 作 者： 组号：**

**实验地点： C5-359**

**实验时间： 2022 年 05 月 04 日 星期 三**

**提交时间： 2022/05/04**

# 实验目的与要求

**实验目的**: 理解TINY语言的词法及词法分析器的实现，并基于该词法分析器，实现拓展语言TINY+的词法分析器。

**实验要求**：

（1） TINY+词法分析器以TINY+源代码为输入，输出为识别出的token序列；

（2） 词法分析器以**最长匹配为原则**，例如‘:=’应识别为赋值符号而非单独的‘：’及‘=’；

（3） Token以（种别码，属性值）表示，包含以下类型的种别码：

a) KEY为关键字；

b) SYM为系统特殊字符；

c) ID为变量；

d) NUM为数值常量；

e) STR为字符串常量。

（4） 识别词法错误。词法分析器可以给出词法错误的行号并打印出对应的出错消息，主要包含以下类型的词法错误：

a) 非法字符。即不属于TINY+字母表的字符，比如$就是一个非法字符；

b) 字符串匹配错误，比如右部引号丢失，如‘scanner

c) 注释的右部括号丢失或匹配错误，如{this is an example

# 实验内容与方法

(1)TINY语言的特证

1、TINY语言无过程，无声明，所有的变量都是整形。

2、它只有两个控制语句：if语句和repeat语句。if语句有一个可选的else部分且必须由关键字end结束。

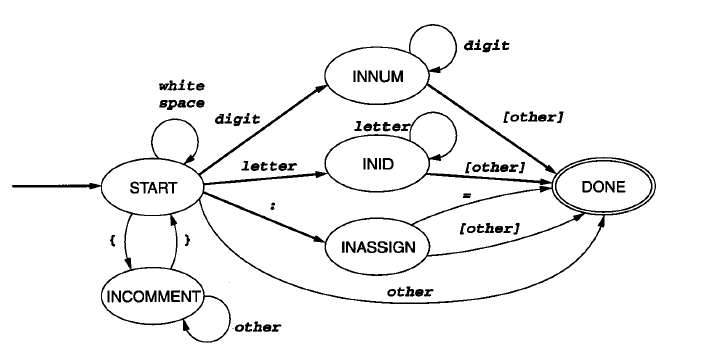
3、read和write完成输入和输出

4、"{"和"}"中的语句为注释，但注释不能嵌套

(2)理解 TINY 语言的词法及词法分析器的实现，并基于该词法分析器， 实现拓展语言 TINY+的词法分析器。

# 实验步骤与过程

（1）对TINY语法和TINY+语法进行分析

①TINY词法

|  |  |  |
| --- | --- | --- |
| 保留字 | 特殊符号 | 其他 |
| if then else end  repeat until  read write | **+ - \* / = <**  **( ) ; :=** | **number**  **identifier** |

②TINY+词法

1. The keywords of the language are the following:

true false or and not int bool float double string do while

if then else end repeat until read write

All keywords are reserved and must be written in lowcase

1. Special symbols are the following:

**> <= >=** != **, '**

**{ } ; := + - \* / ( ) < =**

1. Other tokens are ID, NUM and STRING which are defined by the following regular expressions:

# ID=letter (letter | digit)\*

Identifier is letter followed by letters and digits

NUM=digit digit\*

STRING=**'** any character except **' '**

A STRING is enclosed in brackets **'…'**, any character except **'** can appear in a STRING. A STRING can’t be defined more than a line

letter=a|…|z|A|…|Z

digit=0|…|9

Lower and uppercase letters are distinct

1. White space consists of blanks, newlines and tabs. White space is ignored except that it must separate IDs, NUMs, and keywords
2. Comments are enclosed in curly brackets {…} and cannot be nested. Comments can include more than one line.

（2）对TINY语法和TINY+语法进行分析

①根据TINY+语法和TINY语法的差异，补充完善新字符到GLOBALS.H的TokenType集合中，并声明新状态。

TINY语法的TokenType集合如下：

typedef enum

/\* book-keeping tokens \*/

{

   ENDFILE,

   ERROR,

   /\* reserved words \*/

   IF,

   THEN,

   ELSE,

   END,

   REPEAT,

   UNTIL,

   READ,

   WRITE,

   /\* multicharacter tokens \*/

   ID,

   NUM,

   /\* special symbols \*/

   ASSIGN,

   EQ,

   LT,

   PLUS,

   MINUS,

   TIMES,

   OVER,

   LPAREN,

   RPAREN,

   SEMI

} TokenType;

添加新增字符后的TINY+词法的TokenType集合如下：

typedef enum

{

   ENDFILE,

   ERROR,

   /\* 保留字 \*/

   IF,

   THEN,

   ELSE,

   END,

   REPEAT,

   UNTIL,

   READ,

   WRITE,

   TRUE1,

   FALSE1,

   OR,

   AND,

   NOT,

   INT,

   BOOL1,

   STRING,

   FLOAT,

   DOUBLE,

   DO,

   WHILE,

   /\*多字符标记 \*/

   ID,

   NUM,

   STR,

   /\* 特殊符号 \*/

   ASSIGN,

   EQ,

   LT,

   MT,

   ME,

   LE,

   PLUS,

   MINUS,

   TIMES,

   OVER,

   LPAREN,

   RPAREN,

   SEMI,

   COMMA,

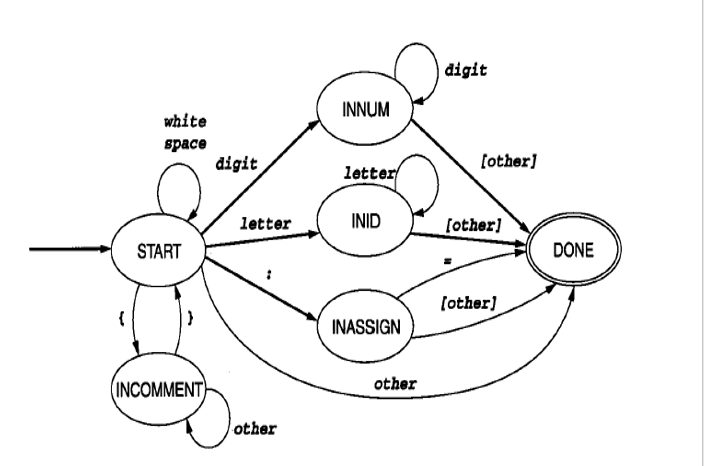
   UPDOX,

   PERCENT

} TokenType;

(3)升级扫描器。

①根据TINY的状态转换图可知，对于TINY词法只要检测到‘<’即结束，但对于TINY+词法需要对‘<=’,‘>=’以及‘<’进行检测判断，因此要新增状态检测新字符。



TINY的字符检测声明

/\* states in scanner DFA \*/

typedef enum

{

  START,

  INASSIGN,

  INCOMMENT,

  INNUM,

  INID,

  DONE

} StateType;

新增字符检测后的TINY+字符检测声明

//枚举类型，DFA状态

typedef enum

{

  START,

  INASSIGN,

  INCOMMENT,

  INNUM,

  INID,

  DONE,

  INLE,

  INME,

  INUPDOX

} StateType;

②修改原状态的操作，补充完善新增状态的操作。

1.更改补充扫描器的第一层，新增INUPDOX状态，同时考虑长度过长、字符串匹配失败、右引号或括号丢失的情况。

    case INUPDOX:

      if (c == '\'')

      {

        save = FALSE;

        state = DONE;

        currentToken = STR;

      }

      else if (!(linepos < bufsize))

      {

        save = FALSE;

        state = DONE;

        currentToken = ERROR;

        strcpy(tokenString, "Missing \" \' \" !");

        tokenStringIndex += 15;

      }

      break;

2.修改扫描器的第二层case,新增对‘,’、‘%’、‘\’这三种字符的检测。

        case ',':

          currentToken = COMMA;

          break;

        case '%':

          currentToken = PERCENT;

          break;

case '\'':

          currentToken = INUPDOX;

          break;

3. 在UTIL.C的printToken()函数中增加新增状态的打印信息。

  case IF:

  case THEN:

  case ELSE:

  case END:

  case REPEAT:

  case UNTIL:

  case READ:

  case TRUE1:

  case FALSE1:

  case OR:

  case AND:

  case NOT:

  case INT:

  case BOOL1:

  case FLOAT:

  case STRING:

  case DOUBLE:

  case DO:

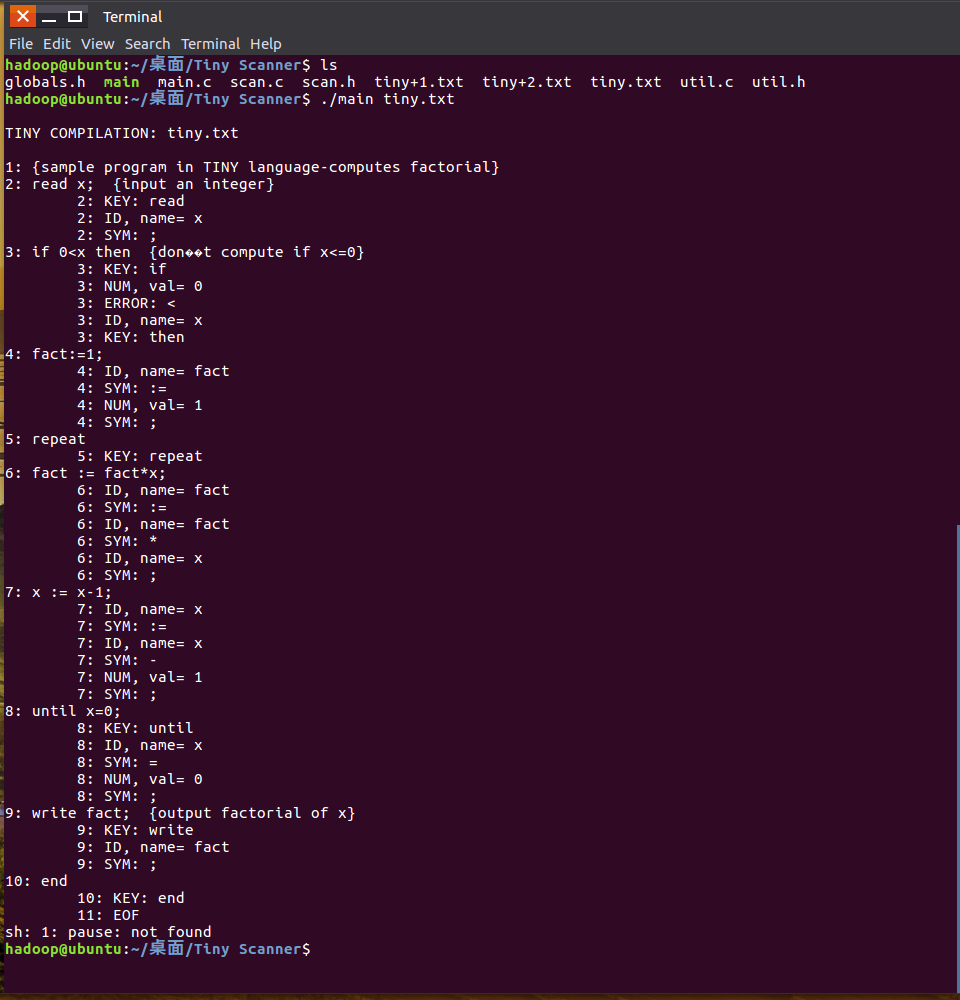
  case STR:

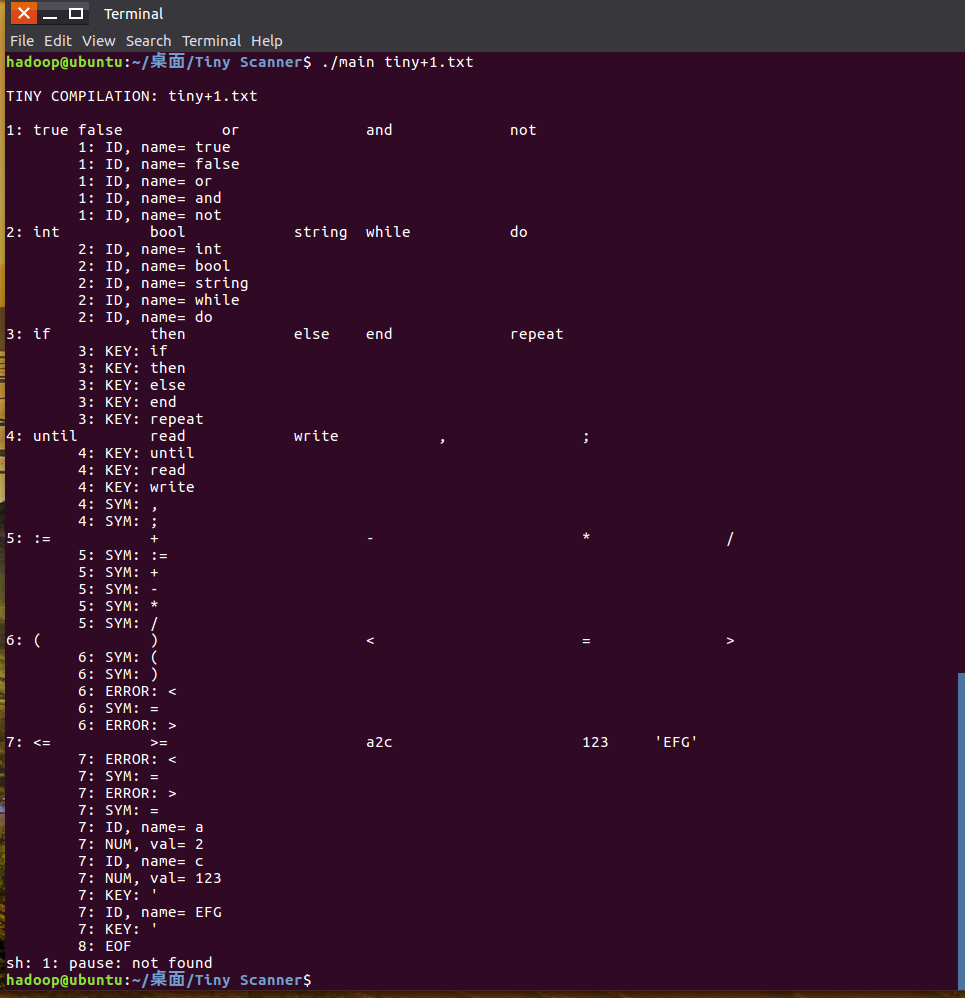
    fprintf(listing,

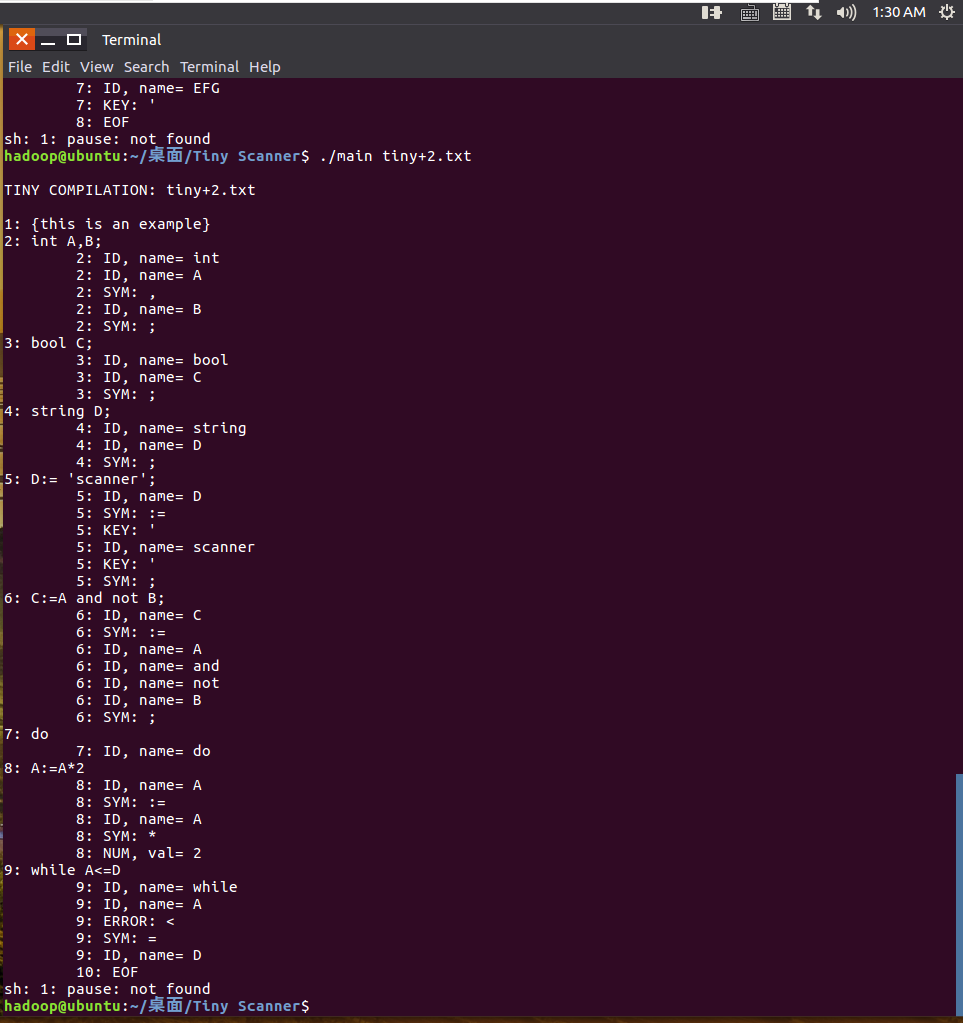
            "STR, val= %s\n", tokenString);

    break;

4.运行测试代码







# 四、实验结论或体会

通过此次TINY+Scanner词法分析程序设计，在读懂代码的过程中进一步理解了理解 TINY 语言的词法及词法分析器的实现，在基于TINY词法分析器实现拓展语言 TINY+的词法分析器的过程中，需要对状态转换图有较好的理解。

|  |
| --- |
| 指导教师批阅意见： |
| 备注： |

注：1、教师批改学生实验报告时间应在学生提交实验报告时间后10日内。