**编译器实践**

**----课程报告**

**1191002008 冯路阳**

# 总体设计

## 编译器目的

目标是实现一个WHI语言的编译器，将高级语言翻译成简单栈式抽象机可识别并执行的中间代码。拟用c语言实现。实现顺序：首先将实现编译器的词法分析部分，在进行调试后依次进行语法分析，代码生成等模块的设计。

## WHI的文法

说明：无符号整型常量（N）、变量（V）、表达式（E）、语句（S）、语句列表（L）、程序（P）

1. P ::= [ X ‘;’ ] L

说明：程序P可以是一个语句列表或者是一个变量定义成分+一个语句列表

1. X ::= ‘var’ V { ‘,’ V }

说明：定义语句X可以是定义一个变量V，也可以是定义多个变量V

1. L ::= S [ ‘;’ L ]

说明：语句列表可以接在语句后，亦可以没有语句列表，仅有语句

1. S ::= ‘skip’ | V ‘:=’ E | ‘read’ ‘(’ V ‘)’ | ‘write’ ‘(’ E ‘)’ | ‘if’ E ‘then’ L ‘else’ L ‘fi’ | ‘while’ E ‘do’ L ‘od’

说明：语句可以是跳过（skip），是变量赋值，是读入（read），是写入（write），是if条件句，是while循环语句

1. E ::= T [ TS ]

说明：表达式E形式多样，我们只讨论他的FIRST集

FIRST(E) = { FIRST(N), FIRST(V), ‘(’, ‘~’}

1. TS ::= ‘+’ T [ TS ] | ‘-’ T [ TS ]

FIRST(TS) = {‘+’, ‘-’}

1. T ::= F [ FS ]

FIRST(T) = { FIRST(N), FIRST(V), ‘(’, ‘~’}

1. FS ::= ‘\*’ F [ FS ] | ‘/’ F [ FS ]

FIRST(FS) = {‘\*’, ’/’}

1. F ::= D [ D1 ]

FIRST(F) = { FIRST(N), FIRST(V), ‘(’, ‘~’}

1. D1 ::= ‘=’ D | ‘<’ D

FIRST(D1) = {‘=’, ‘<’}

1. D ::= N | V | ‘(’ E ‘)’ | ‘~’ E

FIRST(D) = { FIRST(N), FIRST(V), ‘(’, ‘~’}

1. N ::= G { G }

说明：整数N由数字组成

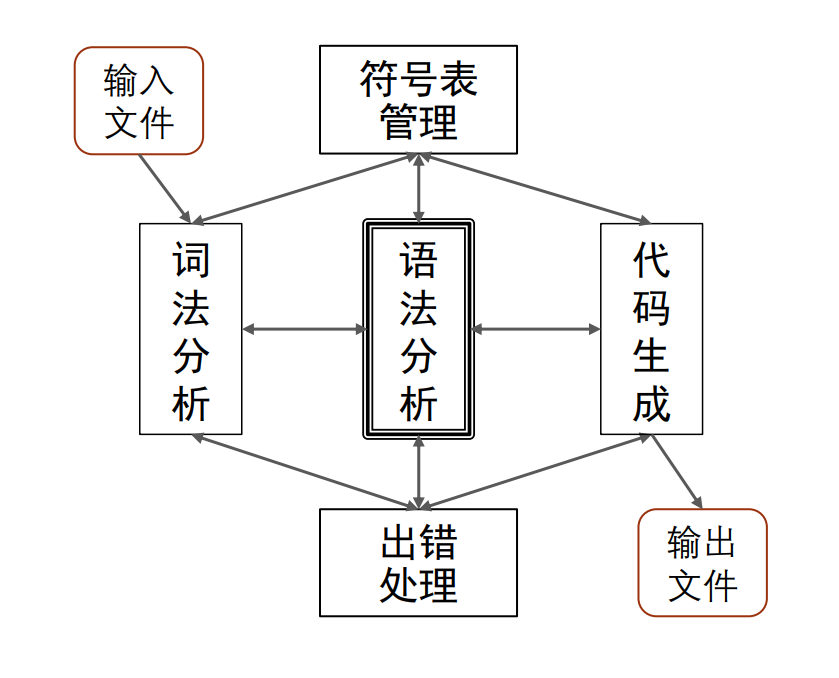
1. V ::= C { C | G }

说明：变量由a-z的字符开头，后可接多个数字或者a-z

1. G ::= ‘0’ | ‘1’ | ‘2’ | ‘3’ | ‘4’ | ‘5’ | ‘6’ | ‘7’ | ‘8’ | ‘9’
2. C ::= ‘a’ | ‘b’ | … | ‘y’ | ‘z’

## 项目实现的工作流程

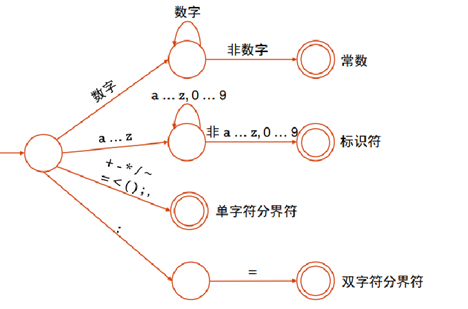
WHI语言🡪单词分析🡪语法分析🡪中间代码生成🡪交给简单栈式抽象机编译成可执行文件



WHI语言编译器实现流程

# 词法分析

## 词法分析器设计



## 词法分析器实现

### 数据与结构设计

宏定义：

#define MAX\_LEN (20) //单词长度限制

#define dummy\_len (10) //保留字每个长度

#define dummy\_num (11) //保留字个数

枚举类：

enum tok\_kind{

    TK\_ID, //这是个变量名

TK\_NUM, //这是个常量

TK\_SEMI, //这是个边界符

TK\_ASSIN, //这是个赋值符号

    TK\_DUMMY, //这是个保留字符

TK\_EOF, //这是个文件结束标志

TK\_ERR //这是个错误字符

};

enum lex\_err{

    err\_char, //错误的单词

err\_len //错误的单词长度

};

结构体：

typedef struct{

    enum tok\_kind kind; //这个单词的种类

    char str[MAX\_LEN]; //这个单词的字符串

    int n\_row; //这个单词在的行号

    int n\_col; //这个单词在的列号

    enum lex\_err err; //这个单词的错误种类

}Token;

全局变量：

int col = 0, row = 1; //col：列 row：行

### 函数设计

#### int overflowed(int n)

功能：判断单词是否超过定长

#### void tk\_print(Token\* tk)

功能：打印处理好的单词，显示相应信息以方便调试

#### int dummy(Token\* tk)

功能：判断一个单词是否为保留关键字

#### Token\* get\_token(FILE \*fp)

功能：将读入的文件进行分词

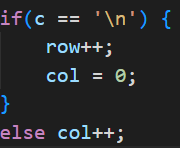
#### Int mian()

功能：主函数

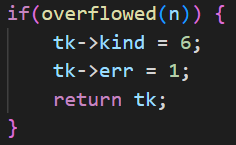
### 关键函数实现细节

#### get\_token()

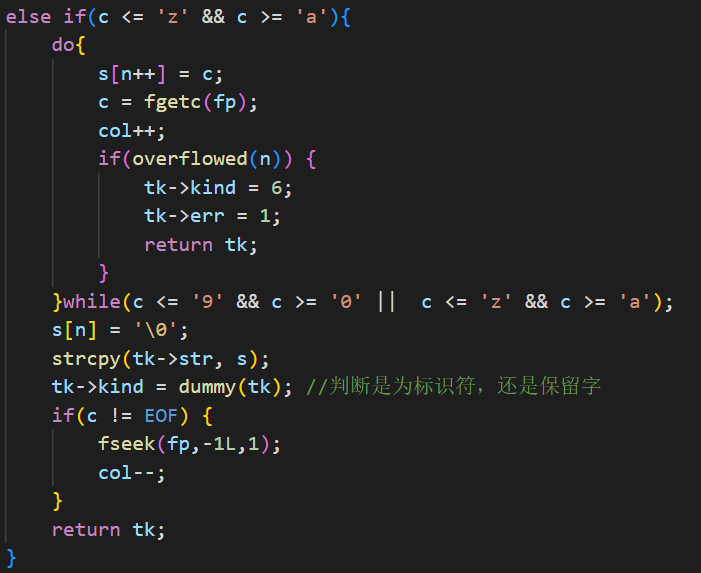
功能（1）：实现全局变量时刻记住当前文件指针的位置，方便出错报告提示



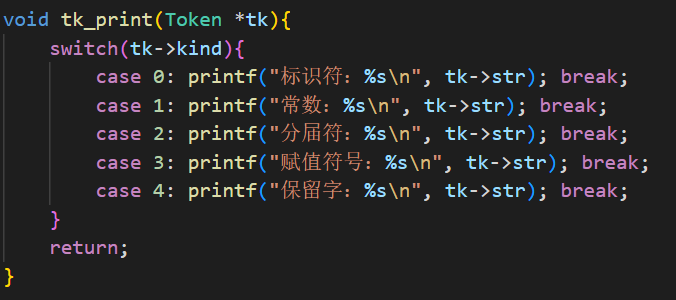
功能（2）：判断输入单词长度是否合法



功能（3）：判断是否为合法的标识符或者是保留字



## 模块调试



解释：用switch语法按需显示分词结果，tk\_kind = 5是文件结束标识。tk\_kind = 6的情况是错误情况，有单独输出如下：

