**＜加法运算符＞ ::= +｜-**

**＜乘法运算符＞  ::= \*｜/**

**＜关系运算符＞  ::=  <｜<=｜>｜>=｜!=｜==**

**＜字母＞   ::= ＿｜a｜．．．｜z｜A｜．．．｜Z**

**＜数字＞   ::= ０｜＜非零数字＞**

**＜非零数字＞  ::= １｜．．．｜９**

**＜字符＞    ::=   '＜加法运算符＞'｜'＜乘法运算符＞'｜'＜字母＞'｜'＜数字＞'**

**＜字符串＞   ::=  "｛十进制编码为32,33,35-126的ASCII字符｝"**

＜程序＞    ::= ［＜常量说明＞］［＜变量说明＞］{＜有返回值函数定义＞|＜无返回值函数定义＞}＜主函数＞

**＜常量说明＞ ::=  const＜常量定义＞;{ const＜常量定义＞;}**

**＜常量定义＞   ::=   int＜标识符＞＝＜整数＞{,＜标识符＞＝＜整数＞}**

**| char＜标识符＞＝＜字符＞{,＜标识符＞＝＜字符＞}**

**＜无符号整数＞  ::= ＜非零数字＞｛＜数字＞｝**

**＜整数＞        ::= ［＋｜－］＜无符号整数＞｜０**

**＜标识符＞    ::=  ＜字母＞｛＜字母＞｜＜数字＞｝**

＜声明头部＞   ::=  int＜标识符＞ |char＜标识符＞

**＜变量说明＞  ::= ＜变量定义＞;{＜变量定义＞;}**

**＜变量定义＞  ::= ＜类型标识符＞(＜标识符＞|＜标识符＞‘[’＜无符号整数＞‘]’){,＜标识符＞|＜标识符＞‘[’＜无符号整数＞‘]’ }**

＜常量＞   ::=  ＜整数＞|＜字符＞

＜类型标识符＞      ::=  int | char

＜有返回值函数定义＞  ::=  ＜声明头部＞‘(’＜参数＞‘)’ ‘{’＜复合语句＞‘}’

＜无返回值函数定义＞  ::= void＜标识符＞‘(’＜参数＞‘)’‘{’＜复合语句＞‘}’

**＜**复合语句＞   ::=  ［＜常量说明＞］［＜变量说明＞］＜语句列＞

＜参数＞    ::= ＜参数表＞

＜参数表＞    ::=  ＜类型标识符＞＜标识符＞{,＜类型标识符＞＜标识符＞}|＜空＞

＜主函数＞    ::= void main‘(’‘)’ ‘{’＜复合语句＞‘}’

＜表达式＞    ::= ［＋｜－］＜项＞{＜加法运算符＞＜项＞}

＜项＞     ::= ＜因子＞{＜乘法运算符＞＜因子＞}

＜因子＞    ::= ＜标识符＞｜＜标识符＞‘[’＜表达式＞‘]’｜＜整数＞|＜字符＞｜＜有返回值函数调用语句＞|‘(’＜表达式＞‘)’

＜语句＞    ::= ＜条件语句＞｜＜循环语句＞| ‘{’＜语句列＞‘}’｜＜有返回值函数调用语句＞;

                      | ＜无返回值函数调用语句＞;｜＜赋值语句＞;｜＜读语句＞;｜＜写语句＞;｜＜空＞;|＜情况语句＞｜＜返回语句＞;

＜赋值语句＞   ::=  ＜标识符＞＝＜表达式＞|＜标识符＞‘[’＜表达式＞‘]’=＜表达式＞

＜条件语句＞  ::=  if ‘(’＜条件＞‘)’＜语句＞［else＜语句＞］

＜条件＞    ::=  ＜表达式＞＜关系运算符＞＜表达式＞｜＜表达式＞ //表达式为0条件为假，否则为真

＜循环语句＞   ::=  for‘(’＜标识符＞＝＜表达式＞;＜条件＞;＜标识符＞＝＜标识符＞(+|-)＜步长＞‘)’＜语句＞

＜步长＞    ::=  ＜非零数字＞｛＜数字＞｝

＜情况语句＞  ::=  switch ‘(’＜表达式＞‘)’ ‘{’＜情况表＞＜缺省＞‘}’

＜情况表＞   ::=  ＜情况子语句＞{＜情况子语句＞}

＜情况子语句＞  ::=  case＜常量＞：＜语句＞

＜缺省＞   ::=  default : ＜语句＞

＜有返回值函数调用语句＞ ::= ＜标识符＞‘(’＜值参数表＞‘)’

＜无返回值函数调用语句＞ ::= ＜标识符＞‘(’＜值参数表＞‘)’

＜值参数表＞   ::= ＜表达式＞{,＜表达式＞}｜＜空＞

＜语句列＞   ::= ｛＜语句＞｝

**＜读语句＞    ::=  scanf ‘(’＜标识符＞{,＜标识符＞}‘)’**

**＜写语句＞    ::= printf ‘(’ ＜字符串＞,＜表达式＞ ‘)’| printf ‘(’＜字符串＞ ‘)’| printf ‘(’＜表达式＞‘)’**

**＜返回语句＞   ::=  return[‘(’＜表达式＞‘)’]**

附加说明：

（1）char类型的表达式，用字符的ASCII码对应的整数参加运算，在写语句中输出字符

（2）标识符区分大小写字母

（3）写语句中的字符串原样输出

（4）情况语句中，switch后面的表达式和case后面的常量只允许出现int和char类型；每个情况子语句执行完毕后，不继续执行后面的情况子语句

（5）数组的下标从0开始

（6）for语句先执行一次循环体中的语句再进行循环变量是否越界的测试