# 编译器课程设计中间代码格式1.0

原则上按照中缀表达式格式输出中间代码，即，形如x = y op z，其中x为结果，y为左操作数，z为右操作数，op为操作符。以下根据基本语法现象举例说明。

* 1. 函数声明

源码形如：

int foo( int a, int b, int c, int d)

中间代码：

int foo()

para int a

para int b

para int c

para int d

* 1. 函数调用

源码形如：

i = tar(x,y)

中间代码：

push x

push y

call tar

i = RET

* 1. 函数返回

源码形如：

return (x)

中间代码：

ret x

* 1. 变量声明

源码形如：

int i, j;

中间代码（符号表信息输出，程序中可不生成真正的中间代码）：

var int i

var int j

* 1. 常数声明

源码形如：

const int c = 10

中间代码（符号表信息输出，程序中可不生成真正的中间代码）：

const int c = 10

* 1. 表达式

源码形如：

x = a \* (b + c)

中间代码（可优化）：

t1 = b + c

t2 = a \* t1

x = t2

* 1. 条件判断

源码形如：

x == y

中间代码：

x == y

* 1. 条件或无条件跳转

中间代码：

goto label1 //无条件跳转到LABEL1

b\_\_\_ label1 //满足条件跳转到LABEL1

(b\_\_可以为bgt, bge, blt, ble, beq, bne)

* 1. 带标号语句

中间代码：

Label\_1 :

x = a + b

* 1. 数组赋值或取值

源码形如：

a[i] = b \* c[j]

中间代码：

t1 = c[j]

t2 = b \* t1

a[i] = t2

* 1. 其他本文档未涉及到的语法现象，或者程序员自行定义的四元式操作，原则上均按照“x = y op z”形式的中缀表达式进行表达。