**实验环境**

**C语言实验环境：Code::Blocks 13.12、GNU GCC Compiler**

**Java语言实验环境：**

**java version "1.8.0\_60"**

**Java(TM) SE Runtime Environment (build 1.8.0\_60-b27)**

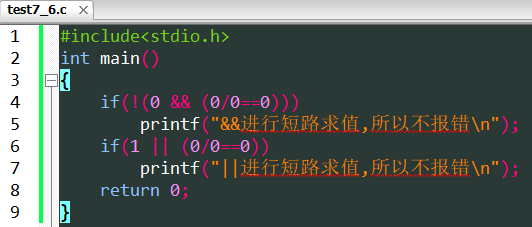
**Java HotSpot(TM) 64-Bit Server VM (build 25.60-b23, mixed mode)**

**文本语言编写：Notepad++**

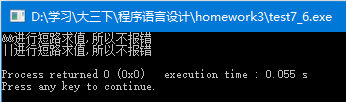
7.6 写一个程序证明短路求值对C/C++的逻辑操作很有用。

解答：

代码编写如下：



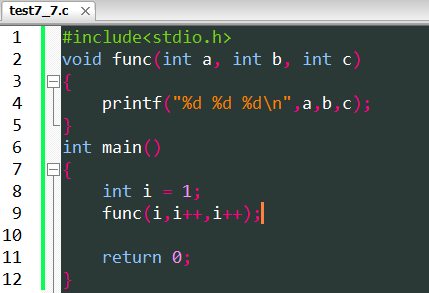
程序运行如下：



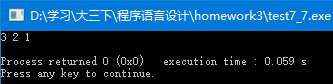
7.7 针对你的C/C++/Ada编译器写一个程序来判定其函数参数的求值次序。

解答：

代码编写如下：



程序运行如下：



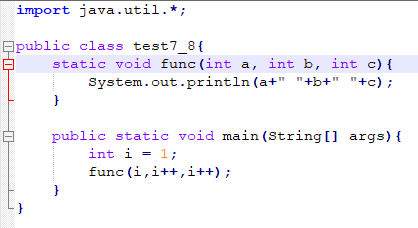
实验结论：

func(i,i++,i++)，输出3 2 1，因为C语言编译器函数参数求值顺序是**“从右至左”**的。

7.8 Java规定所有的表达式，包括调用的参数都是从左到右求值的，为什么Java是这样的，而C/C++不是？

解答：

代码编写如下：



程序运行如下：



实验结果：

Java调用的参数是从左到右求值，可以与7.7题中C语言运行对比。

原因：

C和C ++被设计用于高效执行，而Java旨在实现最小化依赖关系，而不是有效执行，因此Java指定所有表达式（包括调用参数）都从左到右进行求值，而C和C ++不是。

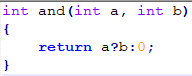
7.9 已经知道在第4章中“+”运算符是左结合的，因此在表达式a+b+c中，先计算表达式a+b，再加上c。而在7.1节中看到，表达式a+b的计算是先处理b，再处理a。二者矛盾吗？这对减法运算符也成立吗？试给出解释。

解答：

不矛盾。

1. “+”的左结合性等价于（a+b）+c，而执行a+b时，先处理b此行为源于表达式求值，与操作符的结合性无关。
2. 对减法运算也成立，因为表达式求值顺序与操作符结合性无关。

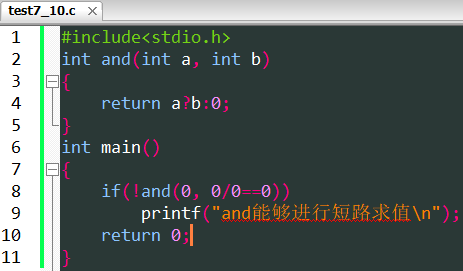
7.10 假定要用C写一个and的短路版本，函数如下:



（a）为什么不正确？（b）如果按范式次序求值，它正确吗？为什么？

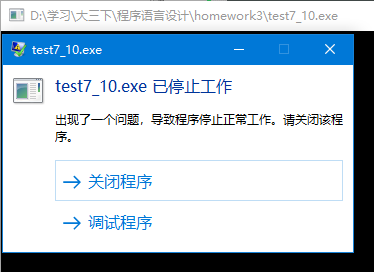
解答：

代码编写如下：



编译运行结果：





回答：

1. “division by zero warning”，当and函数被调用时，传入实参0与0/0==0都会被计算，所以不能进行短路求值。
2. 如果按范式求值，它是正确的，因为参数的计算会按照范式顺序而推迟到它需要被计算时才进行计算。

7.12 考虑表达式：(x != 0) and (y % x == 0) 和 (y % x == 0) and (x != 0)

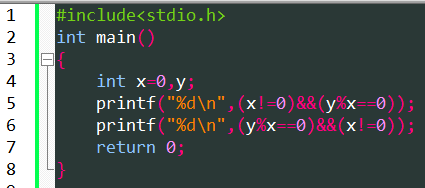
理论上，当x == 0时这两个表达式的值都为false。

(a)在C中，哪个表达式有值？

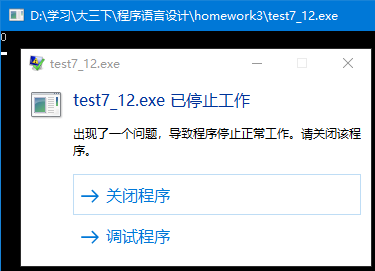
(c)对于某一程序设计语言，这两个表达式是否都可能有值？试予以解释。

解答：

代码编写如下：



编译运行结果：



实验结果：

1. 第一个表达式(x != 0) and (y % x == 0)有值，它的值为0是被成功输出的；第二个表达式产生“devision by zero”的错误是无值无效的。
2. 这两个表达式是都可能有值的。如果我们追求全局优化求值，而不是从左到右的短路求值，那么这两个表达式都能被计算，并且值都为0。