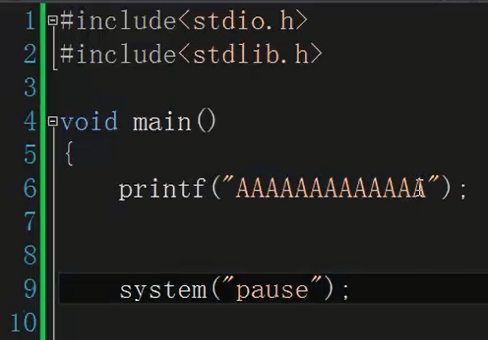
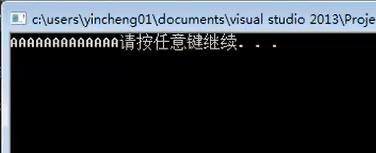
.1.1-5.1.26函数

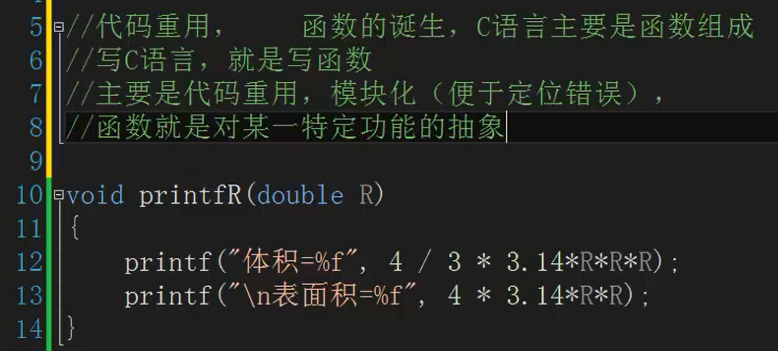
1、 我们先简单调用一个库函数，printf 打印一段字符串



运行结果: (把上面的字符串打印出来)

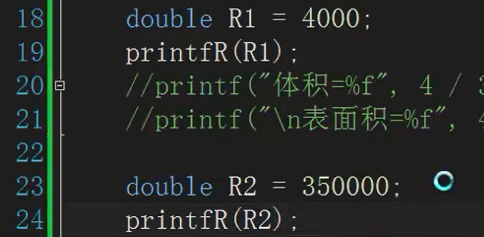


2、定义一个名为printfR的函数



下图是main函数中的 部分代码

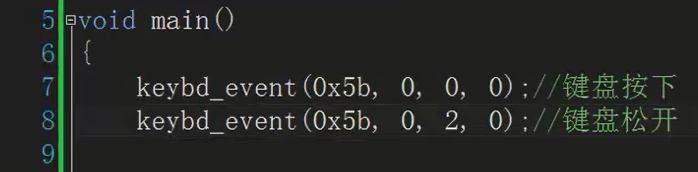
定义R1、R2变量,调用上面的函数printfR函数，把变量R1、R2作为参数传递过去



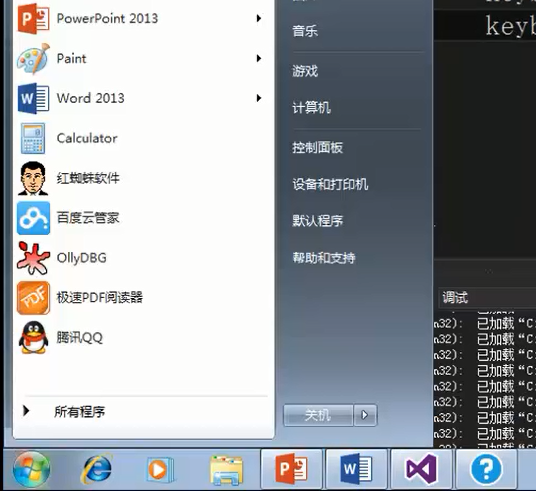
运行结果:



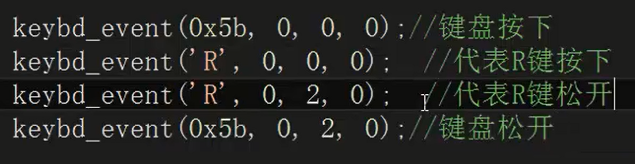
3、模拟键盘操作



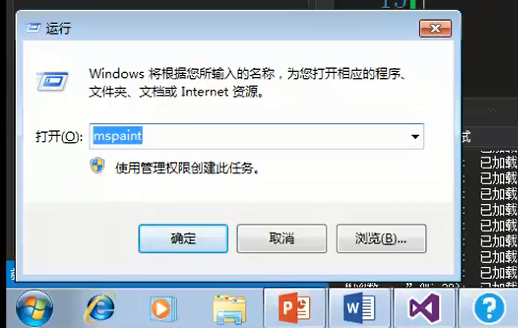
运行结果: 模拟了按下键盘 Win



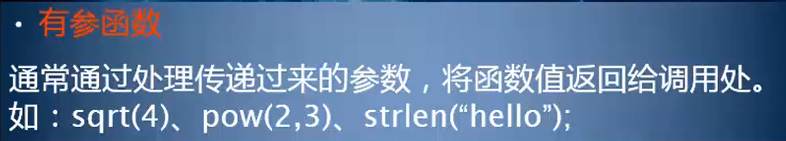
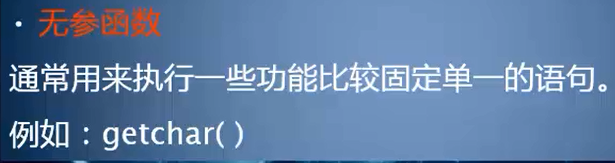
4、模拟键盘操作



运行结果: 模拟了按下 Win+R

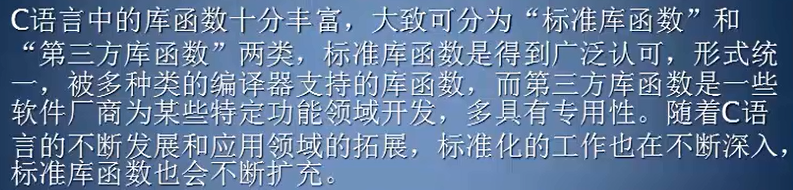


5、函数的分类:有参和无参两种函数

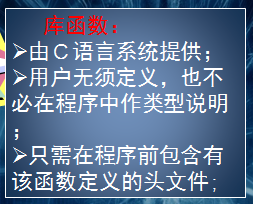


6、库函数:

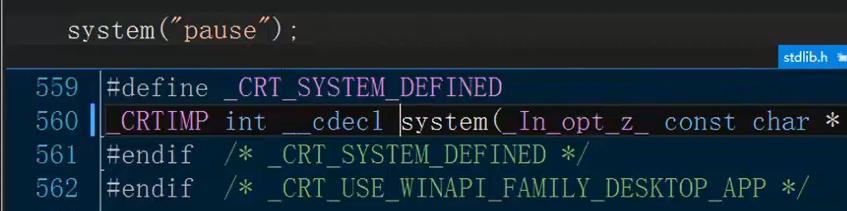
库函数分为 标准库函数 和 第三方库函数



库函数的特点:

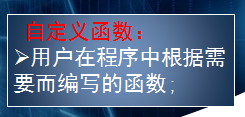
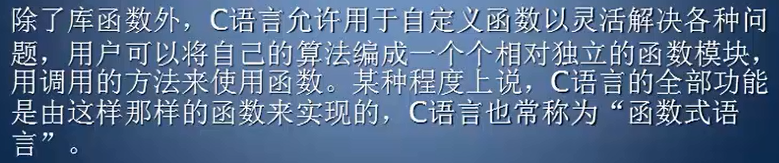


例如 system函数就是 库函数

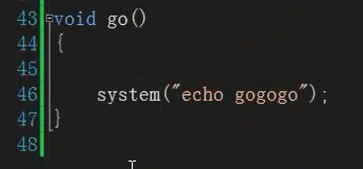


7、自定义函数

用户在程序中自己编写的函数

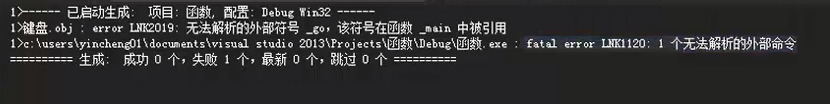


例如 :如下我们定于一个无参函数go



8、当我们编译的时候通常会出现这样一个错误提示:

【 1个无法解析的外部命令 】 出现的 原因通常是因为没有函数的实体



9、函数定义的语法格式:

* 函数定义有4个要素：参数列表，返回类型，函数名和函数体，参数列表和返回类型对应着输入输出，函数名用于和程序中其他程序实体区分，而函数体是一段可执行的代码块，实现特定的算法或功能。



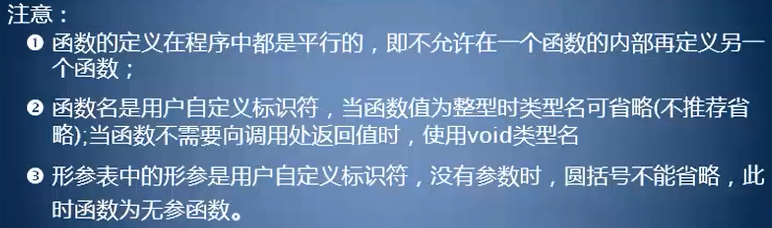
10、函数的调用 有两种类型

定义一个函数是为了调用

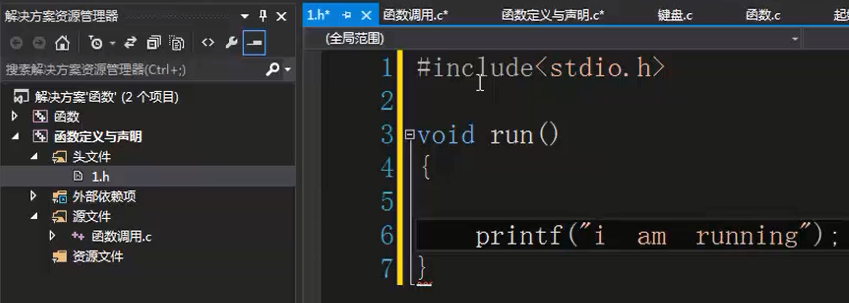
（1）函数调用有两种类型，一是“先定义，后调用”，这要求函数定义和调用语句在同一个文件内，编译器能从函数定义中提取函数的参数列表、输出类型等接口信息。

（2）二是“函数声明+函数调用”，大多数情况下，函数的定义与函数的调用并不在一个文件内，即使在一个文件中也有可能调用在前而定义在后，这时需要在调用之前先对函数声明，告诉编译器有这么一个函数存在，函数原型声明将在后面讨论，下面来看一个先定义、后调用的例子

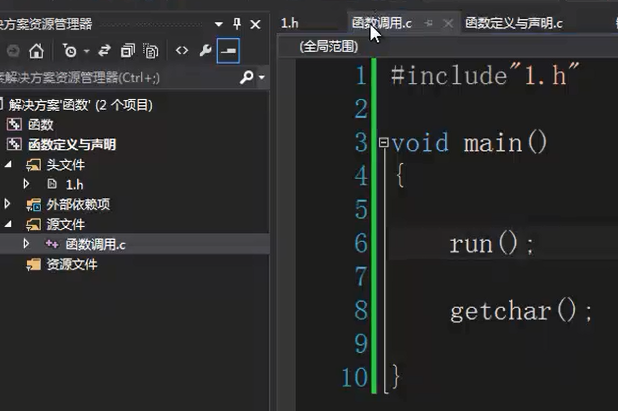
对于函数的主要事项:



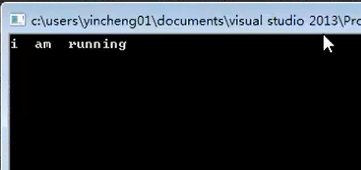
11、自己写一个头文件1.h，并在其中定义一个函数run



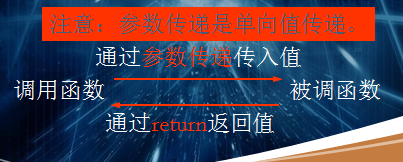
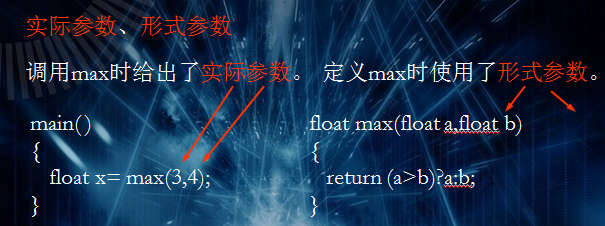
在函数调用.c中我们包含1.h头文件后，调用一下它定义的函数run



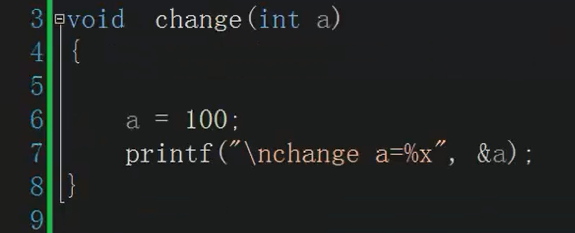
运行结果:



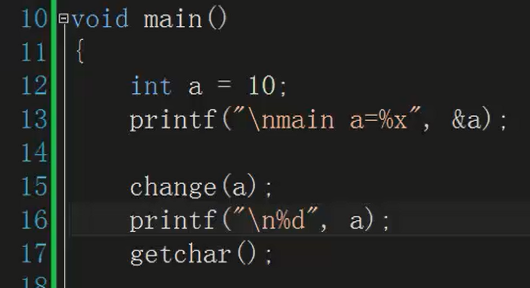
12、实际参数和形式参数



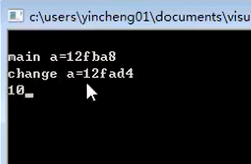
13、我们定义一个函数void change(int a) ;



然后在main函数中调用它，把变量a的值传递给change函数的形参



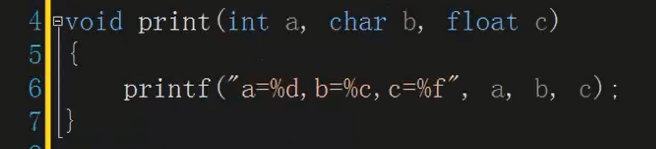
运行结果:



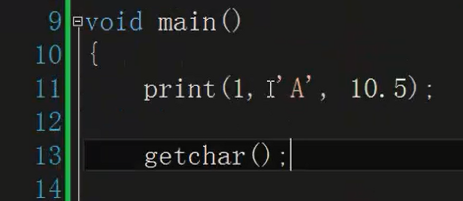
实参a和形参a在内存中的并不是同一个地址。通过调用函数传递值并不能改变

实参的值。

14、定义一个函数print

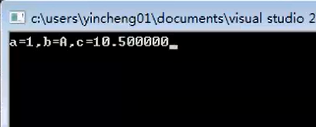


在main函数中调用自定义的函数print



运行结果:

通过定义函数，传递参数，实现数据的打印



15、函数返回值

使用return语句，可以使函数向调用处返回一个值。

它有两个功能:

(1)立即从所在的函数体中退出,返回到调用它的程序中去,

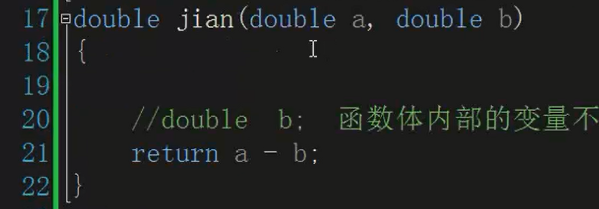
(2)同时返回一个值给调用它的函数。

有两种常用方法可以让函数终止运行并返回到调用它的函数中去:

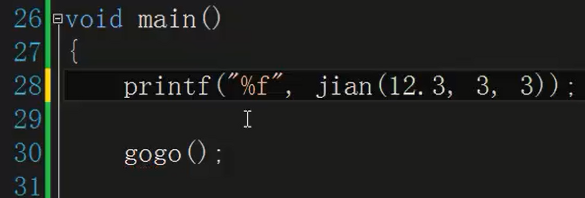
(1)当执行到函数的最后一条语句后返回

(2)当执行到语句return时返回。

定义一个函数jian

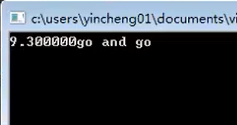


在main函数中调用它

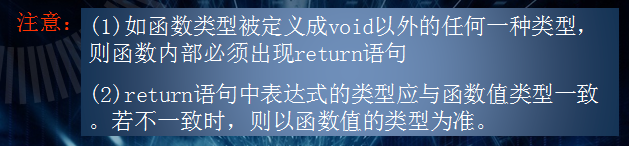


运行结果:

通过调用函数jian(12.3,3.3); 把函数的返回值格式化输出



16、返回值的特点以及注意事项



返回值类型被强制转换

